INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2009; **60**:937–971
Published online 17 October 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1892

A finite point method for adaptive three-dimensional compressible flow calculations

Enrique Ortega*,†, Eugenio Oñate and Sergio Idelsohn‡

International Center for Numerical Methods in Engineering (CIMNE), Universidad Politécnica de Cataluña, Edificio C1, Campus Norte, UPC, Gran Capitán, s/n, 08034 Barcelona, Spain

SUMMARY

The finite point method (FPM) is a meshless technique, which is based on both, a weighted least-squares numerical approximation on local clouds of points and a collocation technique which allows obtaining the discrete system of equations. The research work we present is part of a broader investigation into the capabilities of the FPM to deal with 3D applications concerning real compressible fluid flow problems. In the first part of this work, the upwind-biased scheme employed for solving the flow equations is described. Secondly, with the aim of exploiting the meshless capabilities, an *h*-adaptive methodology for 2D and 3D compressible flow calculations is developed. This adaptive technique applies a solution-based indicator in order to identify local clouds where new points should be inserted in or existing points could be safely removed from the computational domain. The flow solver and the adaptive procedure have been evaluated and the results are encouraging. Several numerical examples are provided in order to illustrate the good performance of the numerical methods presented. Copyright © 2008 John Wiley & Sons, Ltd.

Received 28 January 2008; Revised 27 June 2008; Accepted 13 July 2008

KEY WORDS: meshless methods; finite point method; adaptivity; collocation; compressible flow; time integration explicit

1. INTRODUCTION

Numerical simulation has come into the focus of interest of applied sciences and engineering in the last decades. As a result, the development of numerical techniques for solving partial differential equations (PDEs) has been growing continuously, mainly stimulated by increasing computational

*Correspondence to: Enrique Ortega, International Center for Numerical Methods in Engineering (CIMNE), Universidad Politécnica de Cataluña, Edificio C1, Campus Norte, UPC, Gran Capitán, s/n, 08034 Barcelona, Spain.
†E-mail: eortega@cimne.upc.edu
‡ICREA Research Professor at CIMNE.

Contract/grant sponsor: European Union Programme of High Level Scholarships for Latin America; contract/grant number: E04D027284AR

resources and ever-challenging demands for practical and theoretical applications. Nowadays, there are two main types of numerical techniques for solving PDEs. On the one hand, there exist mesh-based or conventional discretization methods; among them the classical finite differences (FD), finite volume (FV) and finite element (FE) methods are of singular interest. These techniques are mostly employed in practice due to their robustness, efficiency and high confidence gained through years of continuous use and enhancement. On the other hand, there exist meshless methods. Having their *pros* and *cons*, meshless methods offer an alternative to mesh-based techniques. Meshless methods are conceptually attractive; however, their practical implementations have not succeeded so far to prove their efficiency and this is a fact which can explain the comparatively little attention that has been devoted to these techniques. In spite of this, over the last 10 years, some difficulties that arose in conventional mesh-based methods when performing particular applications have brought meshless methods into the focus of attention.

The first meshless methods appeared in the mid-seventies and numerous formulations have been proposed since then. A retrospective view of the evolution of the most relevant meshless methods as well as their connections is presented by Belytschko *et al.* [1]. In their work, the main features of typical meshless methods, their implementation issues and practical applications are offered. An interesting work by Fries and Matthies [2] classifies and analyses the most important meshless methods considering their different origins and viewpoints. The authors highlight the main characteristics and implementation details as well as the advantages and disadvantages of each technique. Some outstanding reviews on meshless methods can also be found in the literature; see for instance those due to Li and Liu [3], Gu [4], Duarte [5], Liu *et al.* [6] and Dolbow and Belytschko [7].

The present work deals with a meshless technique called the finite point method (FPM), which was introduced by Oñate *et al.* [8–10]. In the FPM, the numerical approximation to the problem variables and their derivatives is based on a weighted least-squares (WLSQ) procedure known as fixed least squares (FLS). The strong form of the governing PDEs is sampled at each point by replacing the continuous variables with their approximated counterparts and the resulting system of algebraic equations is obtained by means of a collocation technique.

Since the FPM appeared in the literature towards the mid-nineties, it has been successfully applied to solve convective–diffusive problems, incompressible and compressible fluid flow problems [9–14] and solid mechanics problems [15–17] among others. As regards to fluid flow problems, the first application of the FPM to the solution of the 2D compressible flow equations was presented by Oñate *et al.* [8, 9] and Fischer [12]. In those works, topics such as the construction of local clouds of points and the effects of the weighting function on the numerical approximation were studied using first- and second-order approximation bases. In addition, the compressible flow equations were solved using a Taylor–Galerkin scheme. More recently, Sacco [13] presented a detailed analysis of the finite point (FP) approximation in conjunction with a multi-dimensional application for solving the incompressible flow equations. Outstanding achievements from that work, such as a definition of local and normalized approximation bases, a procedure for constructing local clouds of points as well as a criterion for evaluating their quality, have given the FPM a more solid base. In relation to the solution of the incompressible flow equations, a fractional step algorithm stabilized via a technique known as Finite Calculus (FIC) [18] has also been successfully employed. The FP solution of the 3D compressible flow equations was presented in a pioneer work by Löhner *et al.* [14]. There, two contributions are well worth mentioning: a reliable procedure for constructing the local clouds (based on a Delaunay technique) and a well-suited upwind-biased scheme for solving the flow equations. This scheme is based on a *symmetrized* discrete expression

DOI: 10.1002/fld

of the advective flux-divergence vector, which is composed of a central difference-like expression plus a corrective term. In this scheme, the central difference-like flux term is replaced by an upwind numerical flux obtained through an approximate Riemann solver. In the meshless context, this approach is preferable to artificial dissipation methods as it is not necessary to define any kind of geometrical measure in the cloud of points. Other meshless approaches found in the literature share this philosophy, see for instance [19, 20] and the references cited therein.

All these works, though different, have made remarkable contributions to enhance the performance of the FPM; giving clear evidence of its potential and, in some cases, also revealing important weaknesses. Nowadays, most meshless techniques, and in particular the WLSQ-based methods, are characterized by a lack of solid theoretical and practical arguments regarding local cloud construction, approximation bases selection and weighting function setting, among other important issues. In addition, methods like the FPM, which use the strong form of the differential governing equations, must face some other well-known stability and robustness problems arising from the collocation procedure. Unfortunately, the robustness and the accuracy of the numerical approximation in the cloud of points are dependent on the previously mentioned features. To make matters worse, meshless methods are typically computationally expensive, which requires developing more efficient algorithms and data structures. All these considerations become crucial when dealing with real 3D problems of practical application in engineering. Consequently, improvement in robustness and efficiency seems to be the key to the success of meshless methods in the future.

As regards robustness, some modifications to the FPM have been proposed by Boroomand *et al.* [21] with the aim of reducing instabilities in the minimization procedure, especially those arising from non-appropriate local clouds of points. In addition to that, but from another perspective, we have recently presented an alternative approach towards robustness [22] intended to reduce the local approximation dependence on both the spatial distribution of the cloud of points and the weighting function parameters. This *ad hoc* procedure, which is based on a QR factorization in conjunction with an iterative adjustment of the local approximation parameters, allows obtaining a satisfactory minimization problem solution for cases where usual approaches fail and avoids modifying the geometrical support where the local approximation is based on.

Regardless of the difficulties meshless methods present for practical use, they have potential advantages over conventional discretization techniques, which explain the scientific interest of many researchers in this area (cf. [1–3]). Indeed meshless techniques facilitate the treatment of problems involving moving discontinuities and computational domains whose boundaries change with time and the development of h- and p-adaptivity schemes, among other advantages. In our opinion, these topics constitute key opportunities for the development and promotion of meshless methods.

Along the lines of investigation just mentioned, Perazzo *et al.* [23] have recently presented an h-adaptive technique for solid mechanics problems which is based on the approximation error obtained at each point by the WLSQ functional. In addition, in a previous work [22] we have dealt with high-order FP discretizations in a preliminary manner, exploring the FPM capabilities regarding p-adaptivity. This time, with the same objective in mind, i.e. exploiting the FPM potential, we present an h-adaptive methodology for 2D and 3D compressible flow problems.

The rest of the work is organized as follows. In Section 2, the FP approximation is presented. Section 3 is concerned with the domain discretization and the construction of local clouds of points. Next, in Sections 4 and 5, the upwind-biased scheme employed for solving the 3D Euler equations using the FPM is described. Section 6 provides several numerical calculations to show the performance of the flow solver. Then, an h-adaptive FPM for compressible flow calculations is

DOI: 10.1002/fld

developed in Section 7 and the performance of this adaptive methodology is evaluated by means of several numerical examples in Section 8. Finally, some conclusions of this work are presented in Section 9.

2. NUMERICAL FINITE POINT APPROXIMATIONS ON CLOUDS OF POINTS

In this section, we present an FP approximation to an unknown function $u(\mathbf{x})$ defined in a closed domain $\Omega \in \Re^d$ ($d=1$, 2 or 3), which is discretized by a set of points $\mathbf{x}_i$, $i=1,n$. In order to obtain a local approximation for function $u(\mathbf{x})$, the domain Ω is divided into subdomains Ω_i (henceforth termed *clouds of points*) so that $\Sigma\Omega_i$ represents a covering for Ω. Each local cloud of points consists of a point $\mathbf{x}_i$ called *star point* and a set of points $\mathbf{x}_j$, $j=2,3,\ldots,np$ surrounding $\mathbf{x}_i$, which complete Ω_i. Assuming that function $u(\mathbf{x})$ is smooth enough in Ω_i, it is possible to state the following approximation:

$$u(\mathbf{x}) \cong \hat{u}(\mathbf{x}) = \sum_{l=1}^{m} p_l(\mathbf{x})\alpha_l = \mathbf{p}^{\mathrm{T}}(\mathbf{x})\boldsymbol{\alpha} \tag{1}$$

where $\mathbf{p}(\mathbf{x})$ is a vector whose m-components are the terms of a complete polynomial base in $\Re^d$ (cf. [22] for details) and $\boldsymbol{\alpha}$ is an *a priori* unknown vector. These vectors are given by

$$\begin{aligned} \mathbf{p}_j^{\mathrm{T}} &= [p^1(\mathbf{x}_j) \;\; p^2(\mathbf{x}_j) \;\; \ldots \;\; p^m(\mathbf{x}_j)] \quad (1\times m) \\ \boldsymbol{\alpha} &= [\alpha^1 \;\; \alpha^2 \;\; \ldots \;\; \alpha^m]^{\mathrm{T}} \quad (m\times 1) \end{aligned} \tag{2}$$

Next, at each point $\mathbf{x}_j \in \Omega_i$ the unknown function is obtained as follows:

$$\mathbf{u}^h = \begin{bmatrix} u_1^h \\ u_2^h \\ \vdots \\ u_{np}^h \end{bmatrix} \cong \begin{bmatrix} \hat{u}_1 \\ \hat{u}_2 \\ \vdots \\ \hat{u}_{np} \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1^{\mathrm{T}} \\ \mathbf{p}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{p}_{np}^{\mathrm{T}} \end{bmatrix} \boldsymbol{\alpha} = \mathbf{P}\boldsymbol{\alpha} \tag{3}$$

where $u_j^h = u^h(\mathbf{x}_j)$ is the value of the unknown function $u(\mathbf{x})$ at $\mathbf{x}=\mathbf{x}_j$, $\hat{u}_j = \hat{u}(\mathbf{x}_j)$ is the approximated value at that point and

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{p}_{np}^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} p^1(\mathbf{x}_1) & p^2(\mathbf{x}_1) \ldots p^m(\mathbf{x}_1) \\ & \vdots \\ p^1(\mathbf{x}_{np}) & p^2(\mathbf{x}_{np}) \ldots p^m(\mathbf{x}_{np}) \end{bmatrix} \quad (np\times m) \tag{4}$$

In order to solve the equation system (3) the condition $np=m$ must be fulfilled. This penalizes the approximation flexibility and does not suit a meshless methodology. Thus, $np \geqslant m$ is adopted and the equation system becomes overdetermined. Consequently, an approximate solution is sought by means of a WLSQ technique. This solution minimizes a discrete L_2 error norm in the approximation to $u(\mathbf{x})$ in Ω_i.

DOI: 10.1002/fld

The WLSQ approximation features depend on the shape of the chosen weighting function and the manner in which the latter is applied. In the FPM a fixed weighting function, centred on the star point of the cloud, is chosen so that it satisfies the following conditions:

$$\begin{aligned} \varphi_i(\mathbf{x}_j) &> 0 \quad \forall \mathbf{x}_j \in \Omega_i \\ \varphi_i(\mathbf{x}) &= 0 \quad \forall \mathbf{x} \notin \Omega_i \\ \varphi_i(\mathbf{x}_i) &= 1 \end{aligned} \tag{5}$$

This kind of approximation, known as FLS method, can be considered as a particular case of the moving least-squares (MLS) method introduced by Lancaster and Salkauskas in the context of interpolation and data fitting [24]. When the FLS procedure is applied, the approximation methodology is considerably simplified and its computational cost reduced. It should be noticed, though, that FLS approximations lead to multivalued shape functions depending on the cloud in which the approximation is calculated, i.e. $\mathbf{N}_n(\mathbf{x}_j) \neq \mathbf{N}_m(\mathbf{x}_j)$ (subscripts m and n indicate neighbouring clouds of points). Therefore, the numerical approximation is globally and locally discontinuous and must be considered as valid only at the star point of the cloud where the weighting function is located. Hence, a collocation technique becomes the natural choice in the FPM.

Going back to the minimization procedure, the following discrete functional is defined:

$$\mathrm{J}(\mathbf{x}_i) = \mathrm{J}_i = \sum_{j=1}^{np} \varphi_i(\mathbf{x}_j)[\hat{u}_j - u_j^h]^2 = \sum_{j=1}^{np} \varphi_i(\mathbf{x}_j)[\mathbf{p}_j^{\mathrm{T}}\boldsymbol{\alpha} - u_j^h]^2 \tag{6}$$

in which $\varphi_i(\mathbf{x}_j) = \varphi(\mathbf{x}_j - \mathbf{x}_i)$ is a compact support weighting function. Equation (6) can be rewritten as

$$\mathbf{J} = (\mathbf{P}\boldsymbol{\alpha} - \mathbf{u}^h)^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x})(\mathbf{P}\boldsymbol{\alpha} - \mathbf{u}^h) \tag{7}$$

where $\boldsymbol{\phi}(\mathbf{x}) = \mathrm{diag}(\varphi(\mathbf{x}_j - \mathbf{x}_i))$. The minimization of Equation (7) with respect to $\boldsymbol{\alpha}$ leads to the following equation system:

$$(\mathbf{P}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x})\mathbf{P})\boldsymbol{\alpha} - (\mathbf{P}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}))\mathbf{u}^h = \mathbf{0} \tag{8}$$

known as *normal equations* in the least-squares (LSQ) literature. Introducing the matrices

$$\begin{aligned} \mathbf{A} &= (\mathbf{P}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x})\mathbf{P}), \quad A_{kl} = \sum_{j=1}^{np} \varphi_i(\mathbf{x}_j)\, p_k(\mathbf{x}_j)\, p_l(\mathbf{x}_j) \quad (m \times m) \\ \mathbf{B} &= (\mathbf{P}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x})), \quad B_{lj} = p_l(\mathbf{x}_j)\varphi_i(\mathbf{x}_j) \quad (m \times np) \end{aligned} \tag{9}$$

it is possible to express the normal equations (8) as follows:

$$\mathbf{A}\boldsymbol{\alpha} = \mathbf{B}\mathbf{u}^h \tag{10}$$

As a fixed weighting function is chosen, the unknown coefficients α_j are constant in Ω_i. These coefficients can be found by

$$\boldsymbol{\alpha} = \mathbf{A}^{-1}\mathbf{B}\mathbf{u}^h \tag{11}$$

DOI: 10.1002/fld

Equation (11) must be solved via matrix $\mathbf{A}$ inversion because vector $\mathbf{u}^h$ is not known in advance. Thus, depending on the spatial distribution of the local cloud of points (especially for the 3D case), matrix $\mathbf{A}$ can become ill-conditioned, making it very difficult to invert it with accuracy.

Then, supposing that Equation (11) is solved accurately enough and replacing the coefficients α_j in Equation (1), the approximation to the unknown function at the star point is obtained as

$$\hat{u}(\mathbf{x}_i) = \underbrace{\mathbf{p}^{\mathrm{T}}(\mathbf{x}_i)\mathbf{A}^{-1}\mathbf{B}}_{\mathbf{N}_i^{\mathrm{T}}(\mathbf{x}) \quad (1\times np)}\mathbf{u}^h \tag{12}$$

where $\mathbf{N}_i^{\mathrm{T}}(\mathbf{x}) = [N_{i,1}, N_{i,2}, \ldots, N_{i,np}]$ is the shape function vector of point $\mathbf{x}_i$ in Ω_i. The adoption of an FLS scheme, where matrices $\mathbf{A}$ and $\mathbf{B}$ are constant in Ω_i, simplifies the calculation of the shape functions derivatives. Consequently,

$$\frac{\partial^l \mathbf{N}_i^{\mathrm{T}}(\mathbf{x})}{\partial \mathbf{x}_k^l} = \frac{\partial^l \mathbf{p}^{\mathrm{T}}(\mathbf{x}_i)}{\partial \mathbf{x}_k^l}\mathbf{A}^{-1}\mathbf{B} \tag{13}$$

and the approximation to the unknown function derivatives at $\mathbf{x}_i$ is given by

$$\frac{\partial^l \hat{u}(\mathbf{x}_i)}{\partial \mathbf{x}_k^l} = \frac{\partial^l \mathbf{N}_i^{\mathrm{T}}(\mathbf{x})}{\partial \mathbf{x}_k^l}\mathbf{u}^h = \frac{\partial^l \mathbf{p}^{\mathrm{T}}(\mathbf{x}_i)}{\partial \mathbf{x}_k^l}\mathbf{A}^{-1}\mathbf{B}\mathbf{u}^h \tag{14}$$

The solution of Equations (8) by direct inversion of matrix $\mathbf{A}$ is not the most accurate way of solving the LSQ problem. Thus, it must be restricted to cases when the condition number of matrix $\mathbf{A}$ is moderate. In this work, the procedure adopted to calculate the shape function and its derivatives is the following (cf. [22] for a detailed description). Given a certain cloud of points, first the direct inversion of matrix $\mathbf{A}$ is attempted. If the condition number of $\mathbf{A}$ is smaller than a given maximum admissible value, and if the calculated shape functions satisfy some quality tests, then the shape functions are accepted. If some of the preceding requirements are not met, Equations (8) are solved by an alternative procedure based on QR factorization. The aim of using a QR factorization technique is to get an acceptable solution for cases where the usual procedure fails without having to modify the geometrical structure of the cloud. The WLSQ problem solution via QR factorization may cost, in terms of CPU-time, up to twice as much as the solution via matrix $\mathbf{A}$ inversion if $np \gg m$ [25]. However, this extra amount of time is quite unimportant in the overall time, as the alternative QR-based procedure is only applied to problematic clouds of points that represent only a small percentage of the whole clouds in the domain. The QR factorization-based procedure applied for solving the normal equations system (8) can be summarized as follows:

If matrix $\mathbf{P}$ (given by Equation (4)) has rank m and $np > m$, it can be uniquely factored as

$$\mathbf{P} = \mathbf{Q}\mathbf{R} \tag{15}$$

where matrix $\mathbf{Q} \in \Re^{np\times m}$ is orthogonal ($\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{I}$) and matrix $\mathbf{R} \in \Re^{m\times m}$ is upper triangular with positive diagonal elements (a similar procedure, based on columns pivoting, can be applied for cases where matrix $\mathbf{P}$ is rank deficient or near rank deficient). In order to apply the QR factorization for solving our WLSQ problem, it is necessary to obtain an equivalent unweighted problem. Thus, the next factorization is proposed

$$\tilde{\boldsymbol{\phi}}(\mathbf{x}) = \sqrt{\boldsymbol{\phi}(\mathbf{x})} \quad \text{such that } \tilde{\boldsymbol{\phi}}^{\mathrm{T}}\tilde{\boldsymbol{\phi}} = \boldsymbol{\phi} \tag{16}$$

DOI: 10.1002/fld

and also the following modification of matrix $\mathbf{P}$:

$$\tilde{\mathbf{P}} = \tilde{\boldsymbol{\phi}}\mathbf{P} \tag{17}$$

After that, it is possible to write an equation system equivalent to the one given by Equation (8) as

$$(\tilde{\mathbf{P}}^{\mathrm{T}}\tilde{\mathbf{P}})\boldsymbol{\alpha} = (\tilde{\mathbf{P}}^{\mathrm{T}}\tilde{\boldsymbol{\phi}})\mathbf{u}^{h} \tag{18}$$

Then, the modified matrix (17) is factorized, i.e. $\tilde{\mathbf{P}} = \mathbf{QR}$, and replaced in the equivalent unweighted problem (18). This leads to

$$\mathbf{R}\boldsymbol{\alpha} = \mathbf{Q}^{\mathrm{T}}\tilde{\boldsymbol{\phi}}\mathbf{u}^{h} \tag{19}$$

from which the unknown coefficients α_j can be obtained

$$\boldsymbol{\alpha} = \mathbf{R}^{-1}(\mathbf{Q}^{\mathrm{T}}\tilde{\boldsymbol{\phi}})\mathbf{u}^{h} \tag{20}$$

Here, matrix $\mathbf{R}$ is generally well-conditioned and its inverse is easy to obtain with accuracy, even for the cases when matrix $\mathbf{P}$ is near rank deficient. The described procedure allows us get shape functions of quite good quality in cases where they cannot be obtained via inversion of matrix $\mathbf{A}$. This reduces the dependence of the approximation on the spatial distribution of points and on the functional shape of the weighting function significantly, giving robustness to the FP approximation methodology.

2.1. The weighting function

In the present work, the following normalized Gaussian weighting function is adopted:

$$\varphi_j(x_j) = \frac{\mathrm{e}^{-(d_j/\alpha)^k} - \mathrm{e}^{-(\beta/\alpha)^k}}{1 - \mathrm{e}^{-(\beta/\alpha)^k}} \tag{21}$$

where $d_j = \|\mathbf{x}_j - \mathbf{x}_i\|$, $\alpha = \beta/w$ and $\beta = \gamma d_{\max}$ ($\gamma > 1.0$). The support of this function is isotropic, circular and spherical in two- and three-spatial dimensions, respectively. A detailed description of the effects of the free parameters w, k and γ on the numerical approximation and some guidelines for their setting was presented in [22]. However, an important remark about the parameter γ should be mentioned. The parameter γ determines the size of the weighting function's support and, in consequence, a larger value of γ could be interpreted as an enlargement of the overlapping zone between neighbouring clouds of points. This provides a mechanism for improving the approximation quality where sudden changes in the distance between neighbouring points happen, e.g. near localized adaptive-refined zones and certain details of 3D geometries. In these cases, which generally lead to highly distorted clouds of points, good results are obtained setting $1 < \gamma < 1.25$.

3. DISCRETIZATION OF THE DOMAIN AND LOCAL CLOUD CONSTRUCTION

An adequate support of points is essential for setting a good local approximation for each cloud. Even though the iterative QR-based technique described above attempts to reduce this dependence, the spatial support of the approximation continues playing a major role. At present, there is not a

DOI: 10.1002/fld

unique criterion to determine the size, shape and structure of the local spatial support and several procedures have been proposed by meshless practitioners. Concerning the FPM, an appropriate methodology for constructing local clouds of points (based on a Delaunay technique) has been suggested by Löhner *et al.* [14]. In the present work we follow the general criteria proposed there.

3.1. Domain discretization

The point discretization of the analysis domain Ω is obtained by means of a modification of the algorithm presented in [26]. It starts from a Delaunay triangulation that bounds the domain and inserts new points in the centre of empty spheres filling Ω. This incremental quality technique, known as *optimization driven point insertion*, allows achieving a fast point discretization of the analysis domain well-suited for FP calculations.

3.2. Local cloud construction

The local clouds of points are constructed as follows. Given a point discretization of the computational domain and a set of normal vectors belonging to the triangulation that bounds this domain, a maximum ($np_{\max}$) and minimum ($np_{\min}$) allowable number of points in the cloud and an initial search radius are set. Then, for each star point $\mathbf{x}_i$, all neighbours within the search radius (r_s) are found through an octree technique. Any local cloud of points inside the computational domain is constructed with the closest neighbouring points from the star point. However, if a star point $\mathbf{x}_i$ is located either over or close enough to a solid boundary, the points included in its cloud (admissible points) must also satisfy the conditions described below.

Case 1: Star point located over a solid boundary.

In this particular case (sketched in Figure 1), every point $\mathbf{x}_j$ located within the search radius is admissible if it meets the following conditions:

$$\cos(\theta) \geqslant \cos(\pi/2+\delta), \quad \cos(\theta) = \frac{\mathbf{n}_i \cdot \mathbf{r}_j}{\|\mathbf{n}_i\| \|\mathbf{r}_j\|} \tag{22}$$

$$\|\mathbf{r}_j^t\| < \alpha r_{\text{search}} \tag{23}$$

Condition (22) defines an admissible zone around the start point, which is defined in the normal direction to the surface and δ is a small angle dependent on the surface curvature. The second condition (23) imposes a certain aspect ratio in the cloud, given by the parameter $\alpha \neq 0$.

Case 2: Cloud of points intercepting a solid boundary.

In this case the point $\mathbf{x}_j$ located over a surface ($\mathbf{x}_{j_{\text{nea}}}$), nearest to the star point $\mathbf{x}_i$, must be sought (see Figure 1). Then, every point within the search radius is admissible if

$$\cos(\theta) \geqslant \cos(\pi/2+\delta), \quad \cos(\theta) = \frac{\mathbf{n}_{j_{\text{nea}}} \cdot \mathbf{r}_j}{\|\mathbf{n}_{j_{\text{nea}}}\| \|\mathbf{r}_j\|} \tag{24}$$

and no restriction is imposed to the aspect ratio of the cloud of points.

If the number of admissible points found within the search radius is not enough, the latter is increased until condition $np_{\min} \leqslant np \leqslant np_{\max}$ is satisfied. Otherwise, if the number of admissible points goes beyond $np_{\max}$, only the $np_{\max}$ points nearest to $\mathbf{x}_i$ are added to the cloud.

It is very helpful to force the first layer of Delaunay nearest neighbours of $\mathbf{x}_i$ into the local cloud of points when sudden variations in the distance between neighbouring points occur inside the analysis domain. For each star point this is accomplished by performing a local Delaunay grid

DOI: 10.1002/fld

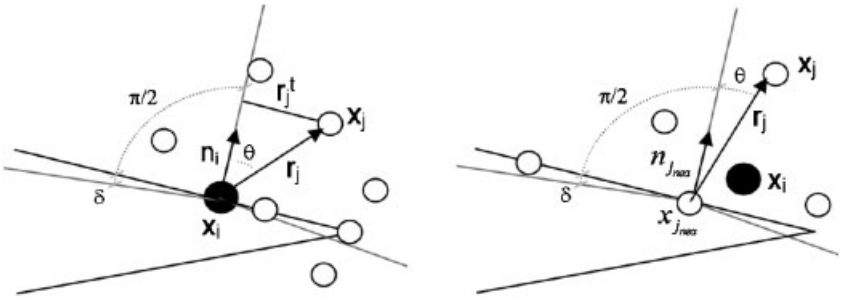

Figure 1. The construction of local clouds near the boundaries. Left: the star point located over a solid boundary and right: a cloud of points intercepting a solid boundary.

with all the points falling within the octree search area. Only the first layer of nearest neighbours is retained and used to initialize the local cloud of points. Finally, admissible nearest points are added until the condition $np_{\min} \leqslant np \leqslant np_{\max}$ is fulfilled. This procedure, that follows the lines proposed by Löhner *et al.* [14], avoids non-overlapping neighbouring clouds of points and improves the quality of the local discretization. Furthermore, the information concerning the first layer of neighbouring points for each star point is useful for improving several computational procedures. In the present work such information is needed for the adaptive procedure presented in Section 7.

4. THE EULER EQUATIONS

The first-order hyperbolic system of Euler equations can be written in several equivalent forms. Their conservative differential form is given by

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}^k}{\partial x_k} = \mathbf{0} \tag{25}$$

where $k = 1, d$ being d the number of spatial dimensions of the problem. $\mathbf{U}$ is the conservative variables vector and $\mathbf{F}^k$ is the advective flux vector in the spatial direction x_k. These vectors are defined as

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u_i \\ \rho e_t \end{bmatrix}, \quad \mathbf{F}^k = \begin{bmatrix} \rho u_k \\ \rho u_i u_k + \delta_{ik} p \\ (\rho e_t + p) u_k \end{bmatrix} \tag{26}$$

where ρ, p and e_t, respectively, denote the density, pressure and total energy of the fluid; u_i is the i-component of the velocity vector, δ_{ik} is the Kronecker delta and subscripts $i, k = 1, d$. The following state relation for a perfect gas closes the system of equations (25)

$$p = \rho(\gamma - 1)[e_t - \tfrac{1}{2} u_i u_i] \tag{27}$$

in which $\gamma = C_p / C_v$ is the specific heats ratio (in the present work we adopt $\gamma = 1.4$).

The solution of Equation (25) in a closed domain $\Omega \in \Re^d$ with boundaries $\Gamma = \Gamma_\infty \cup \Gamma_w$ requires appropriate initial and boundary conditions. The initial conditions only start the explicit calculation

DOI: 10.1002/fld

and they are simple to implement. In general, they could be taken from the far-field state $\mathbf{U}_\infty$. Regarding the boundary conditions, those employed in the present work are of two different kinds. The first one is concerned with *far-field conditions* applied on the outer boundaries Γ_∞ and the second one is concerned with *slip wall conditions* applied on the solid boundaries Γ_w. In the case of far-field boundary conditions, the prescribed fluxes at each boundary point are obtained solving an approximate Riemann problem in the outward normal direction to the boundary, between the boundary point state $\mathbf{U}_i$ and the far-field state $\mathbf{U}_\infty$. Over solid boundaries, slip wall conditions are applied forcing the fluxes to remain tangent to the boundaries, i.e. cancelling their components in the boundary normal direction.

5. THE FLOW SOLVER

In this section, the numerical strategy adopted for solving the compressible flow equations is set forth. Despite some modifications to the way in which the divergence of the advective fluxes is discretized in the local cloud of points, the overall scheme follows the general lines proposed by Löhner *et al.* [14].

Recalling the FPM approximation procedure described in Section 2, for each star point $\mathbf{x}_i \in \Omega$ we can write the following approximations:

$$\hat{\mathbf{U}}(\mathbf{x}_i) = \hat{\mathbf{U}}_i = \sum_{j \in \Omega_i} N_{ij} \mathbf{U}_j^h$$
$$\hat{\mathbf{F}}^k(\mathbf{x}_i) = \hat{\mathbf{F}}_i^k = \sum_{j \in \Omega_i} N_{ij} (\mathbf{F}_j^k)^h \tag{28}$$

where $N_{ij} = N_i(\mathbf{x}_j)$ is the shape function of the star point $\mathbf{x}_i$ evaluated at the cloud's point $\mathbf{x}_j$ and $(\mathbf{F}_j^k)^h = \mathbf{F}^k(\mathbf{U}_j^h)$. Then, the one-dimensional semi-discrete counterpart of Equation (25) can be expressed for each star point $\mathbf{x}_i$ by

$$\frac{\partial \hat{\mathbf{U}}_i}{\partial t} = -\frac{\partial \hat{\mathbf{F}}_i}{\partial x} = -\sum_{j \in \Omega_i} \frac{\partial N_{ij}}{\partial x} \mathbf{F}_j^h = -\sum_{j \in \Omega_i} b_{ij} \mathbf{F}_j^h \tag{29}$$

where $\mathbf{F}_j^h$ is the advective flux vector calculated at a point $\mathbf{x}_j \in \Omega_i$ and the coefficient b_{ij} stands for the shape function derivative of $\mathbf{x}_i$ evaluated at the same point $\mathbf{x}_j$.

It is important to note that the $(\cdot)^h$ parameters do not coincide with the approximated ones $(\hat{\cdot})$ because in the FP method the shape functions do not interpolate point data. These values are related by Equation (28), which implies that a linear system must be solved in order to get the $(\cdot)^h$ parameters. Fortunately, this equation system has excellent properties and can be solved by a few iterations of a Gauss–Seidel method or similar. Henceforth, the markers $(\hat{\cdot})$ and $(\cdot)^h$ will be omitted for the sake of simplicity.

Taking advantage of the partition of nullities property of the shape function derivatives it is possible to infer

$$\sum_{j \in \Omega_i} b_{ij} = b_{ii} + \sum_{j \neq i} b_{ij} = 0 \;\rightarrow\; b_{ii} = -\sum_{j \neq i} b_{ij} \tag{30}$$

DOI: 10.1002/fld

Replacing Equation (30) in Equation (29), the following semi-discrete expression is obtained:

$$\frac{\partial \mathbf{U}_i}{\partial t} = -\sum_{j \neq i} b_{ij}(\mathbf{F}_j - \mathbf{F}_i) \tag{31}$$

Equation (31) is unstable and needs to be stabilized. For that purpose, a more suitable equivalent form is sought scaling by a factor of $\frac{1}{2}$ the stencil of points [20] used for its calculation. In this way, we obtain a totally equivalent semi-discrete expression, which is given by

$$\frac{\partial \mathbf{U}_i}{\partial t} = -2\sum_{j \neq i} b_{ij}(\mathbf{F}_{ij} - \mathbf{F}_i) \tag{32}$$

where $\mathbf{F}_{ij}$ is an *a priori* unknown numerical flux vector, evaluated at the midpoint of the line segment connecting the star point $\mathbf{x}_i$ with another point $\mathbf{x}_j \in \Omega_i$ (see Figure 2). Many possibilities for calculating $\mathbf{F}_{ij}$ can be found in the literature. Following the ideas presented in [14], the Roe's approximate Riemann solver [27] is adopted in this work. Then, the numerical flux results

$$\mathbf{F}_{ij} = \tfrac{1}{2}(\mathbf{F}_j + \mathbf{F}_i) - \tfrac{1}{2}|\mathbf{A}(\mathbf{U}_i, \mathbf{U}_j)|(\mathbf{U}_j - \mathbf{U}_i) \tag{33}$$

where $\mathbf{A}(\mathbf{U}_i, \mathbf{U}_j)$ is the flux Jacobian matrix evaluated at the Roe average-state between the points $\mathbf{x}_i$ and $\mathbf{x}_j$, i.e. $\mathbf{U}_L = \mathbf{U}_i$ and $\mathbf{U}_R = \mathbf{U}_j$. In order to calculate the absolute value of the Roe matrix the procedure suggested by Turkel [28] is applied. This procedure avoids costly matrix–matrix and matrix–vector multiplications in the calculation of the dissipative term $|\mathbf{A}(\mathbf{U}_i, \mathbf{U}_j)|(\mathbf{U}_j - \mathbf{U}_i)$.

The multi-dimensional extension of the scheme presented above is straightforward. For each pair of points $(\mathbf{x}_i, \mathbf{x}_j)$, a one-dimensional problem is solved in the direction of vector $\mathbf{l}_{ji} = \mathbf{x}_j - \mathbf{x}_i$ to obtain the midpoint numerical flux $\mathbf{F}_{ij}$. Then, $\mathbf{F}_{ij}$ is projected onto the Cartesian axis and the semi-discrete scheme (32) results

$$\frac{\partial \mathbf{U}_i}{\partial t} = -2\sum_{j \neq i} b_{ij}^k [\mathbf{F}_{ij}^k - \mathbf{F}_i^k] \tag{34}$$

where $k = 1, d$ being d the number of spatial dimensions of the problem. The Cartesian components of the midpoint numerical flux are obtained by

$$\mathbf{F}_{ij}^k = \tfrac{1}{2}(\mathbf{F}_j^k + \mathbf{F}_i^k) - \tfrac{1}{2}|\mathbf{A}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j)|(\mathbf{U}_j - \mathbf{U}_i) \cdot \hat{n}^k \tag{35}$$

where $\hat{\mathbf{n}}$ is a versor in the direction of the vector $\mathbf{l}_{ji}$ and $|\mathbf{A}_{\hat{n}}(\mathbf{U}_i, \mathbf{U}_j)|$ denotes the absolute value of the Roe matrix calculated in the same direction. The stencil of points employed in the derivation of expression (34) is presented in Figure 3.

5.1. Increasing spatial accuracy

The low-order scheme we have developed is useless in practice. In order to make this scheme suitable for capturing all the flow features with precision, it is necessary to increase its spatial

Figure 2. The one-dimensional stencil of points.

DOI: 10.1002/fld

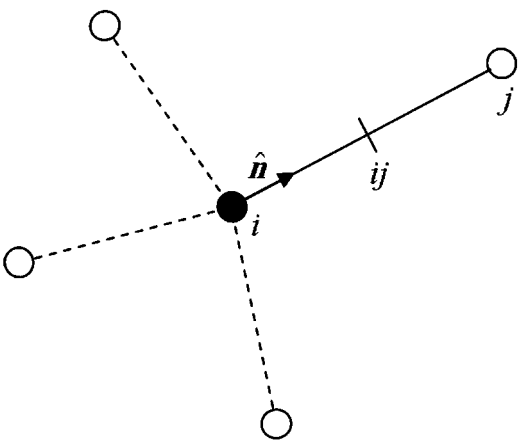

Figure 3. The multi-dimensional stencil of points.

order of accuracy. This is accomplished by replacing the zero-order extrapolation of the variables ($\mathbf{U}_L = \mathbf{U}_i$ and $\mathbf{U}_R = \mathbf{U}_j$) at the midpoint $\mathbf{x}_{ij}$ by a higher-order extrapolation. The Monotone Upstream-centered Schemes for Conservation Laws (MUSCL) methodology [29] allows achieving accurate second- and third-order schemes using linear and quadratic reconstruction of the variables, respectively. Unfortunately, this high-order methodology does not guarantee an oscillation-free solution and monotonicity should be enforced by introducing non-linear *limiters* into the reconstruction process. In brief, these limiters recognize any local extrema of the solution field and automatically switch, at these points, the high-order extrapolation to a zero-order extrapolation, avoiding the appearance of under and overshoots in the numerical solution.

Taking into consideration the high-order approach proposed in [14], in this work we adopt a MUSCL reconstruction of the variables in conjunction with the Van Albada limiter. This results in the following set of reconstructed variables:

$$\begin{aligned} \mathbf{U}_i^+ &= \mathbf{U}_i + \frac{\mathbf{s}_i}{4}[(1-\eta)(\mathbf{U}_i - \mathbf{U}_{i-1}) + (1+\eta)(\mathbf{U}_j - \mathbf{U}_i)] \\ \mathbf{U}_j^- &= \mathbf{U}_j - \frac{\mathbf{s}_j}{4}[(1-\eta)(\mathbf{U}_{j+1} + \mathbf{U}_j) + (1+\eta)(\mathbf{U}_j - \mathbf{U}_i)] \end{aligned} \tag{36}$$

where $\mathbf{U}_i^+$ and $\mathbf{U}_j^-$ are, respectively, the leftward and rightward extrapolations to the conservative variables vector at point $\mathbf{x}_{ij}$. In the above expressions the choice of the parameter $\eta = -1$ leads to a second-order, leftward-biased scheme for $\mathbf{U}_i$ and a rightward-biased scheme for $\mathbf{U}_j$. For $\eta = 1$ and $\eta = \frac{1}{3}$, a second-order centred scheme and a third-order scheme are obtained, respectively. The Van Albada limiters $\mathbf{s}_i$ and $\mathbf{s}_j$ are given by [14]

$$\begin{aligned} \mathbf{s}_i &= \max\left[0, \frac{2(\mathbf{U}_i - \mathbf{U}_{i-1})(\mathbf{U}_j - \mathbf{U}_i) + \varepsilon}{(\mathbf{U}_i - \mathbf{U}_{i-1})^2 + (\mathbf{U}_j - \mathbf{U}_i)^2 + \varepsilon}\right] \\ \mathbf{s}_j &= \max\left[0, \frac{2(\mathbf{U}_{j+1} - \mathbf{U}_j)(\mathbf{U}_j - \mathbf{U}_i) + \varepsilon}{(\mathbf{U}_{j+1} - \mathbf{U}_j)^2 + (\mathbf{U}_j - \mathbf{U}_i)^2 + \varepsilon}\right] \end{aligned} \tag{37}$$

DOI: 10.1002/fld

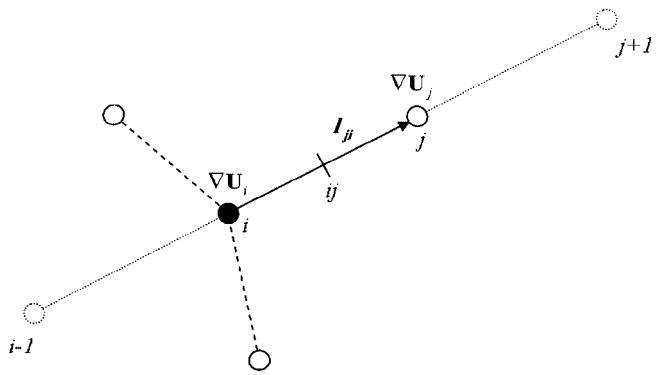

Figure 4. Implementation of the multi-dimensional reconstruction of the variables.

where $\varepsilon \approx 1.0\text{E}-5$ is a small constant included to avoid divisions by zero. The variables $\mathbf{U}_{i-1}$ and $\mathbf{U}_{j+1}$ are obtained by a centred approximation to the $\nabla\mathbf{U}$ at the points i and j

$$\begin{aligned} \mathbf{U}_i - \mathbf{U}_{i-1} &= 2\mathbf{l}_{ji} \cdot \nabla\mathbf{U}_i - (\mathbf{U}_j - \mathbf{U}_i) \\ \mathbf{U}_{j+1} - \mathbf{U}_j &= 2\mathbf{l}_{ji} \cdot \nabla\mathbf{U}_j - (\mathbf{U}_j - \mathbf{U}_i) \end{aligned} \tag{38}$$

in which $\mathbf{l}_{ji} = \mathbf{x}_j - \mathbf{x}_i$ is the vector linking the points i and j (see Figure 4).

Once the high-order extrapolations (36) have been calculated, the midpoint numerical flux (35) is modified according to

$$\mathbf{F}_{ij}^k = \tfrac{1}{2}(\mathbf{F}^k(\mathbf{U}_i^+) + \mathbf{F}^k(\mathbf{U}_j^-)) - \tfrac{1}{2}|\mathbf{A}_{\hat{n}}(\mathbf{U}_i^+, \mathbf{U}_j^-)|(\mathbf{U}_j^- - \mathbf{U}_i^+) \cdot \hat{n}^k \tag{39}$$

and then, replacing Equation (39) in Equation (34) the high-order semi-discrete scheme is obtained.

5.2. Time discretization

Following the ideas in [14], the temporal discretization of Equation (34) is done in a fully explicit manner by means of a multi-stage method that is a subset of the Runge–Kutta family of schemes. Assuming that the vector of conservative variables $\mathbf{U}^h$ is known at time $t = t^n$, the right-hand side of Equation (34) is calculated for each point ($\mathbf{RHS}_i$). Then, it is possible to advance the solution in time from t^n to t^{n+1} by means of the following *s-stage* scheme:

$$\begin{aligned} \mathbf{U}_i^{(0)} &= \mathbf{U}_i^n \\ &\vdots \\ \mathbf{U}_i^{(s)} &= \mathbf{U}_i^n + \alpha_s \Delta t_i \mathbf{RHS}_i^{(s-1)} \\ &\vdots \\ \mathbf{U}_i^{n+1} &= \mathbf{U}_i^{(s_{\max})} \end{aligned} \tag{40}$$

DOI: 10.1002/fld

where Δt_i is the time-step evaluated at the star point $\mathbf{x}_i$ and α_s are integration coefficients that depend on the number of stages employed ($s_{\max}$). For two-, three- and four-stages schemes these parameters are set as follows:

$$\text{2-stages} \rightarrow \alpha_1 = \tfrac{1}{2} \text{ and } \alpha_2 = 1.0$$
$$\text{3-stages} \rightarrow \alpha_1 = \tfrac{3}{5},\ \alpha_2 = \tfrac{3}{5} \text{ and } \alpha_3 = 1.0$$
$$\text{4-stages} \rightarrow \alpha_1 = \tfrac{1}{4},\ \alpha_2 = \tfrac{1}{3},\ \alpha_3 = \tfrac{1}{2} \text{ and } \alpha_4 = 1.0$$

The difference between the $(\cdot)^h$ parameters and the approximated ones $(\hat{\cdot})$ has already been pointed out in Section 5. Taking into account that $\mathbf{RHS}_i = f(\mathbf{U}_j^h)\nabla \mathbf{x}_j \in \Omega_i$, the following linear system has to be solved at the end of each integration stage:

$$\mathbf{M}\mathbf{U}^h = \hat{\mathbf{U}} \tag{41}$$

where $\mathbf{M} \in \Re^{n \times n}$ is the *mass matrix* of the system, which results from the assembly of the N_{ij} coefficients (see Equation (28)). Fortunately, as mentioned before, this system has excellent properties and can be solved by a few iterations of a Gauss–Seidel method or similar.

It should be noticed that, even though the numerical scheme presented in this section is intended to solve the inviscid compressible flow equations, with minor modifications the same scheme can be applied for solving the viscous flow equations.

6. NUMERICAL EXAMPLES

In this section, some 3D compressible flow calculations are presented with the aim of illustrating the performance of the proposed methodology. The first example concerns a subsonic flow past a sphere. Although this example has barely any practical interest, it allows assessing the low Mach number behaviour of the scheme as well as evaluating its intrinsic dissipation. Then, a transonic flow around the ONERA M6 wing is solved. This example, which is a classic CFD validation test for external flows, allows demonstrating the applicability of the present methodology to practical aerodynamics problems. With the same objective in mind, by the end of this section another transonic flow calculation concerning an NACA wing-body configuration is presented.

6.1. Subsonic flow around a sphere

In this example, subsonic inviscid flow past a sphere is solved for a freestream Mach number $M_\infty = 0.2$. The computational domain is discretized by a non-structured distribution of 30 013 points and second-order spatial approximations are obtained in clouds of points with $30 \leqslant np \leqslant 40$. Next, coefficient of pressure (C_p) and Mach number isolines on the sphere are shown in Figure 5. The calculated C_p distribution around the sphere (in the streamwise direction) is compared with analytical potential flow results in Figure 6. Good agreement between the numerical and potential results can be observed. Note that the separation point on the sphere, obtained by the FP calculation, is almost coincident with the potential rear stagnation point. This fact gives a cue of the low inherent dissipation of the proposed numerical scheme. The point discretization over the sphere and over a cut along the symmetry plane of the computational domain is shown in Figure 7.

DOI: 10.1002/fld

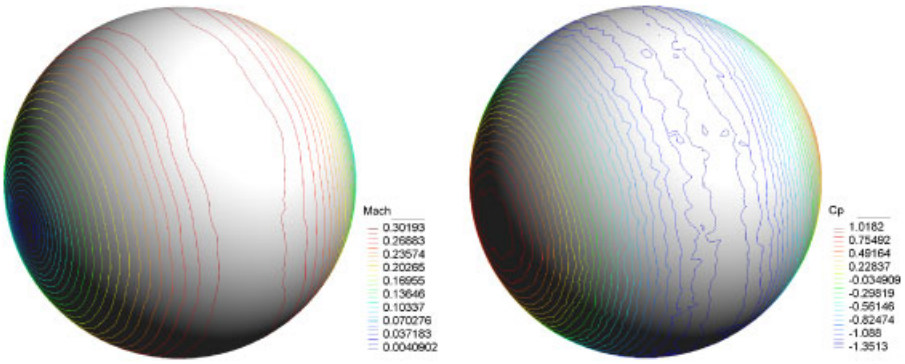

Figure 5. Mach number and C_p isolines on the sphere, $M_\infty = 0.2$.

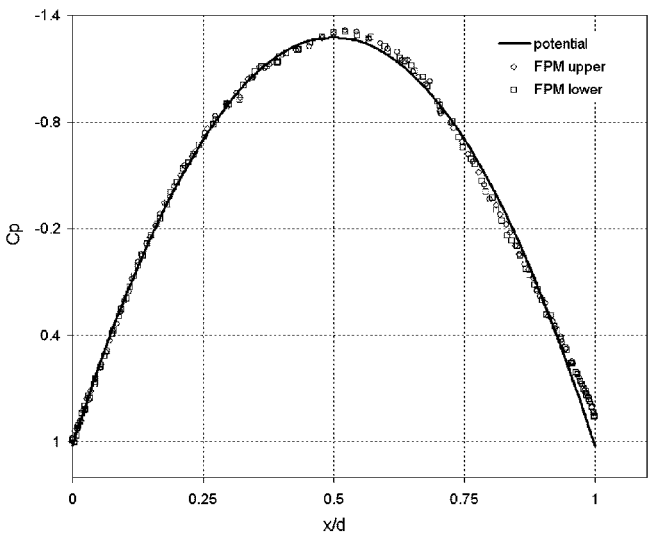

Figure 6. C_p distribution around the sphere; a comparison between the FP calculation and the analytical potential solution. $M_\infty = 0.2$.

6.2. *Transonic flow over the ONERA M6 wing*

This validation test [30] was developed by ONERA in 1972 with the objective of providing experimental support for studies regarding transonic flows at high Reynolds numbers. Since then, these experimental results, which cover a wide range of subsonic and transonic flows, have turned into a classical reference data for code validation assessments. The ONERA M6 is a semi-span wing with a sweepback $\Lambda_{LE} = 30°$, an aspect ratio $A = 3.8$ and a taper ratio $\lambda = 0.562$. The wing-section is an ONERA 'D' symmetrical airfoil constant along the span and the wing has not geometrical twist. In this example we solve the test case # 2308 (cf. [30]) which concerns transonic flow over the ONERA M6 wing set at an incidence angle $\alpha = 3.06°$. The freestream Mach number is

DOI: 10.1002/fld

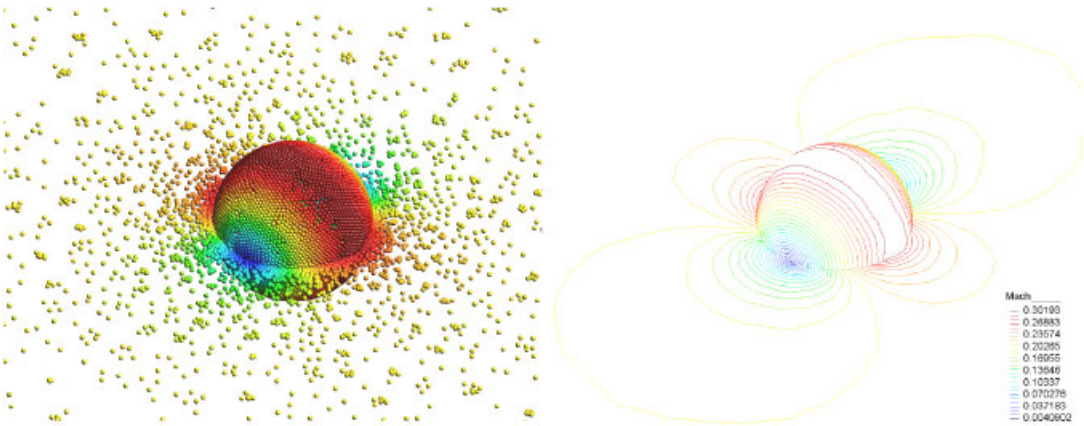

Figure 7. The sphere and the symmetry plane of the problem. Left: points displaying Mach number results and right: Mach number isolines. $M_\infty = 0.2$.

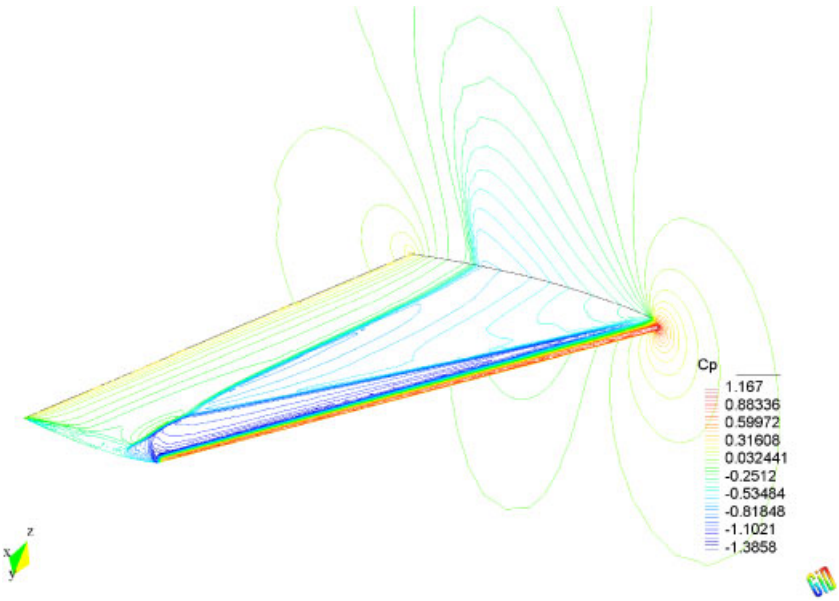

Figure 8. C_p isolines on the upper surface of the ONERA M6 wing and the symmetry plane. $M_\infty = 0.84$ and $\alpha = 3.06°$.

$M_\infty = 0.84$ and the Reynolds number is $Re = 11.7\text{E}6$. The most relevant data about this test case can also be found in [31].

Owing to the fact that in the present work we are solving the Euler equations, our simulation assumes the fluid to be inviscid. The computational domain is discretized by an unstructured

DOI: 10.1002/fld

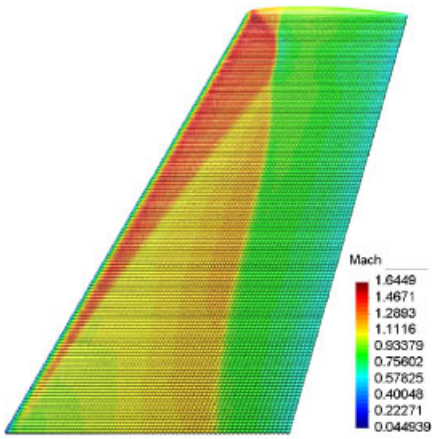

Figure 9. Surface discretization of the ONERA M6 wing (upper surface view); coloured points display Mach number values. $M_\infty = 0.84$ and $\alpha = 3.06°$.

distribution of 512 141 points and second-order approximation bases are employed for calculating the shape functions and their derivatives in clouds with $30 \leqslant np \leqslant 45$. Next, C_p and Mach number numerical results are shown in Figures 8 and 9, respectively.

A comparison between numerical and experimental C_p distributions along several sections on the wing is shown in Figure 10. In accordance with the available experimental data [30], these sections are located at the following spanwise stations: $\eta = 0.2$, 0.44, 0.65, 0.8, 0.9, 0.95 and 0.99 being $\eta = 2y/b$. A good agreement between computed and experimental results can be observed in Figure 10 and, as it was expected, the inviscid computation gives a shock wave, which is slightly stronger than the true shock wave and is located close behind the latter. Notice that the experimental data measured at $\eta = 0.99$ reveals separated flow behind the shock wave on the upper side of the wing. Consequently, experimental and calculated C_p distributions do not match in the separated flow region.

6.3. Transonic flow over an NACA wing-body configuration

This example involves the computation of an inviscid transonic flow over a wing-body configuration [32]. The wing has a sweepback $\Lambda_{1/4} = 45°$, an aspect ratio $A = 4$, a taper ratio $\lambda = 0.6$ and it has not geometrical twist; moreover, the wing-section is an NACA 65A006 airfoil constant along the wing span. The fuselage has a circular cross-section and its rear part is attached to a sting, which supports the model in the wind tunnel test section.

The numerical calculation presented here regards a freestream Mach number $M_\infty = 0.9$ and the model incidence angle is $\alpha = 4°$. The discretization of the computational domain consists of an unstructured distribution of 512 553 points and second-order approximations are built on clouds with $35 \leqslant np \leqslant 45$. Next, C_p and Mach number results computed for the proposed flow conditions are presented in Figures 11 and 12, respectively.

Figure 13 shows a comparison of C_p distributions calculated at two spanwise stations $\eta = 0.4$ and $\eta = 0.8$ on the wing with experimental measurements [32]. Additionally, the longitudinal

DOI: 10.1002/fld

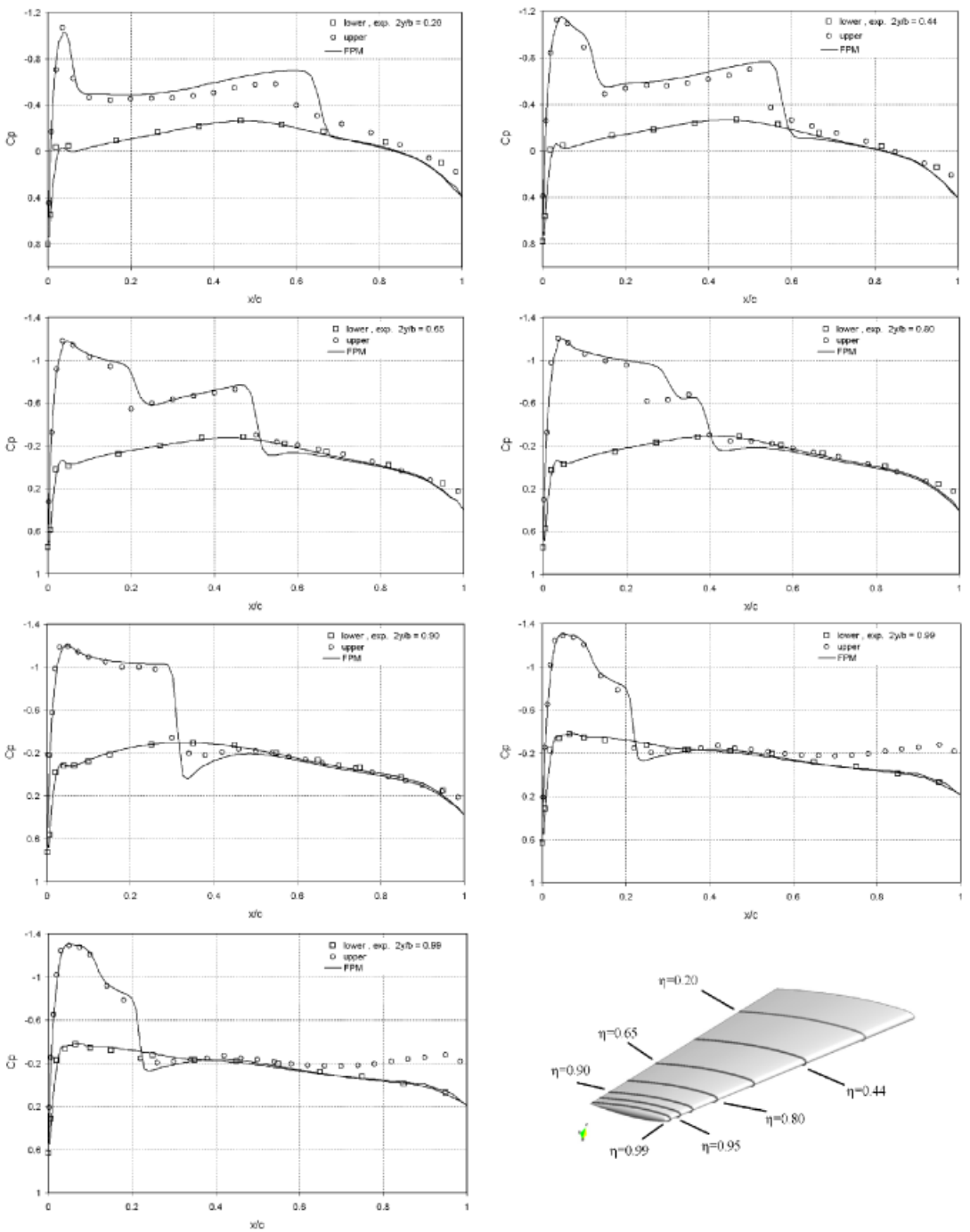

Figure 10. Comparisons between computed and experimental C_p distributions along several sections on the wing. ONERA M6 wing, $M_\infty = 0.84$ and $\alpha = 3.06°$.

DOI: 10.1002/fld

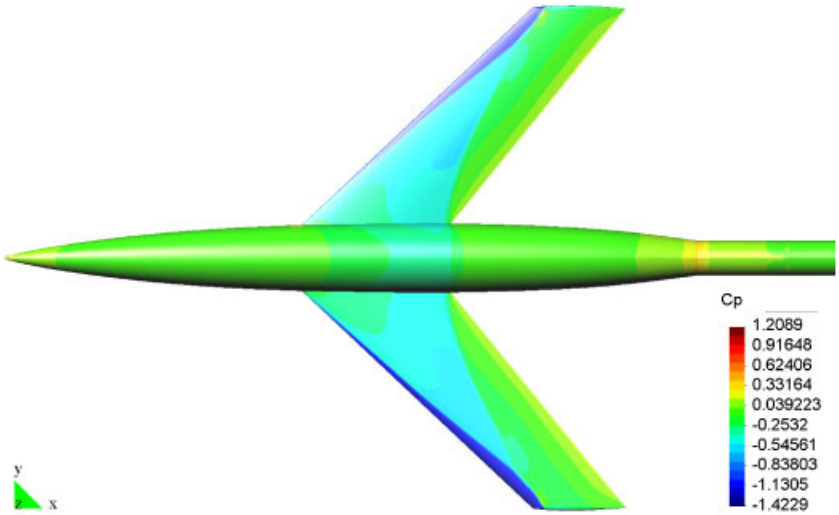

Figure 11. C_p distribution on the NACA wing-body configuration (only half of the model has been calculated, the other part is simply included for visualization purposes). $M_\infty = 0.90$ and $\alpha = 4.0°$.

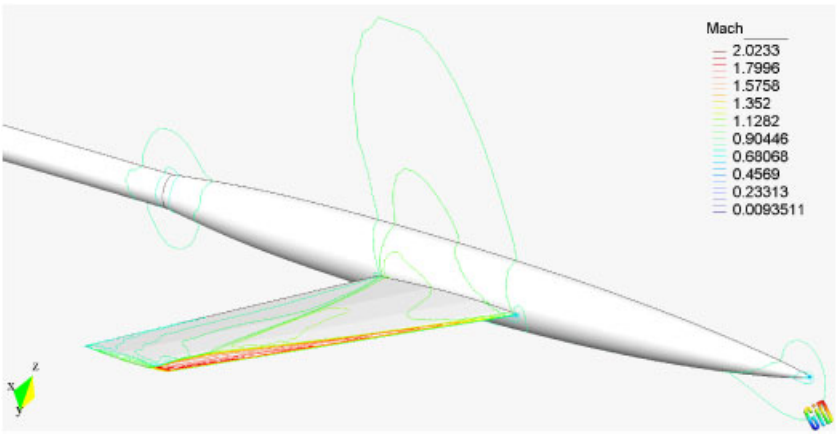

Figure 12. Mach number isolines on the NACA wing-body configuration and the symmetry plane. $M_\infty = 0.90$ and $\alpha = 4.0°$.

C_p distribution along the fuselage symmetry plane is compared with experimental results in Figure 14.

As in the previous case, minor differences (due to the inviscid assumption adopted for the computational flow model) exist between numerical and experimental results. In spite of this, both results match very well as it can be observed in Figures 13 and 14.

DOI: 10.1002/fld

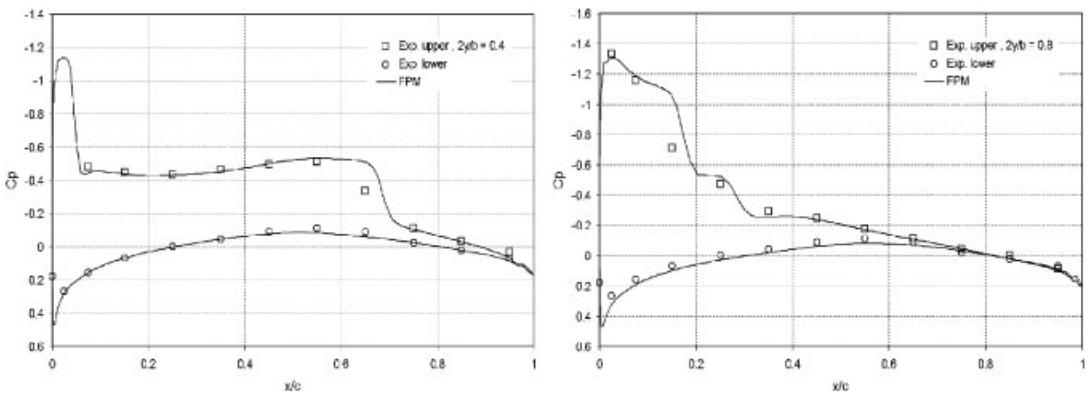

Figure 13. A comparison between computed and experimental C_p distribution along two spanwise wing stations $\eta=0.4$ and 0.8. NACA wing-body configuration, $M_\infty=0.90$ and $\alpha=4.0°$.

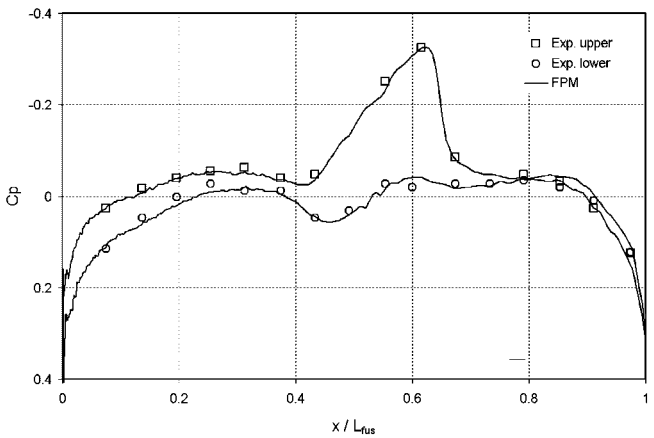

Figure 14. Comparison between computed and experimental C_p distribution along the fuselage symmetry plane. NACA wing-body configuration, $M_\infty=0.90$ and $\alpha=4.0°$.

7. AN h-ADAPTIVE PROCEDURE FOR FP CALCULATIONS

There are several reasons that explain the appeal of mesh (or point) adaptive strategies in the different fields of numerical simulation. Adaptivity reduces the effort needed to obtain a proper discretization for numerical analysis as regards man-hours, CPU-time and memory requirements significantly. In addition, adaptive procedures make the accurate computation of the smaller scales of the flow field easier, especially when we do not have *a priori* information concerning the solution, and become essential for non-stationary problems involving moving discontinuities.

In the introduction to this work we have referred to some topics in numerical computation where meshless approaches seem to have certain advantages over mesh-based approaches and

DOI: 10.1002/fld

adaptivity is one of them. The fact that meshless techniques do not need to keep a conforming mesh makes them specially well-suited for implementing adaptive procedures. With the purpose of exploiting this capability, in this section we develop an adaptive FP procedure for compressible flow problems.

7.1. The refinement criterion

The FP solution at a previous time-step is employed with the aim of identifying local clouds of points where new points should be inserted or existing points could be removed from the computational domain. This is accomplished by a normalized indicator that evaluates, in an approximate manner, the *curvature* of the solution at each point

$$\varphi_i = \frac{1}{\varphi_m} \sum_{j=1}^{nn} |\mathbf{l}_{ji} \cdot (\nabla \rho_j - \nabla \rho_i)|, \quad \varphi_m = \max(\varphi_i), \quad i = 1, n \tag{42}$$

In the expression above *nn* is the number of points in the first layer of Delaunay nearest neighbours of $\mathbf{x}_i$ (already obtained in the local cloud construction stage), $\mathbf{l}_{ji} = \mathbf{x}_j - \mathbf{x}_i$ is the vector linking each pair of points $(\mathbf{x}_i, \mathbf{x}_j)$ and ρ is the density of the fluid. Naturally, another flow variable or a combination of flow variables can be adopted for calculating the refinement indicator (42). The last option could be appropriate for the treatment of viscous fluid flows.

The refinement criterion is applied as follows. Based on Equation (42), new points are inserted around $\mathbf{x}_i$ when $\varphi_i > \varphi_{\max}$ and, conversely, point $\mathbf{x}_i$ is removed from the computational domain if $\varphi_i < \varphi_{\min}$. The limits $\varphi_{\max}$ and $\varphi_{\min}$ depend on the problem under consideration; in the numerical examples presented here $\varphi_{\max} \approx 0.1$ and $\varphi_{\min} \approx 0.005$ are chosen. It should be noticed that in particular cases, the proposed normalization causes a lack of sensitivity to relative small gradients in the flow field. When this happens, it could be useful to avoid the normalization by setting $\varphi_m = 1$ or taking another local maximum for normalizing the indicator.

7.2. The strategy

Once the refinement criterion has been applied, the remaining of the proposed adaptive procedure can be reduced to three main steps: the *insertion* of new points, the *removal* of existing points and an *update*. The latter makes reference to the construction of the data associated with each new point and the re-construction of the data associated with the *affected* existing points, respectively. We consider that an existing point is affected when a new point falls inside its cloud, or the spatial position of any point in its cloud changes due to smoothing.

7.2.1. Insertion of new points. When a star point $\mathbf{x}_i$ is marked to refine ($\varphi_i > \varphi_{\max}$), its Delaunay grid of nearest neighbours is used to calculate the Voronoi vertices surrounding $\mathbf{x}_i$. Next, new *candidate* points $\mathbf{x}_c$ are set at these vertices, i.e. at the centre of the empty circumcircle/circumsphere calculated for each triangle/tetrahedron (2D/3D) composing the Delaunay grid of nearest neighbours. Each candidate point $\mathbf{x}_c$ is accepted if it meets the following requirements:

$\mathbf{r}_1$. The radius of the empty circumcircle/circumsphere (r_c) complies with $r_c > r_{\min}$, being $r_{\min}$ a user-defined parameter which stands for the minimum admissible distance between points.

$\mathbf{r}_2$. The radius r_c is smaller than a certain internal measure (d_e) of the triangle/tetrahedron from which the empty circumcircle/circumsphere originates. The internal measure d_e is calculated as

DOI: 10.1002/fld

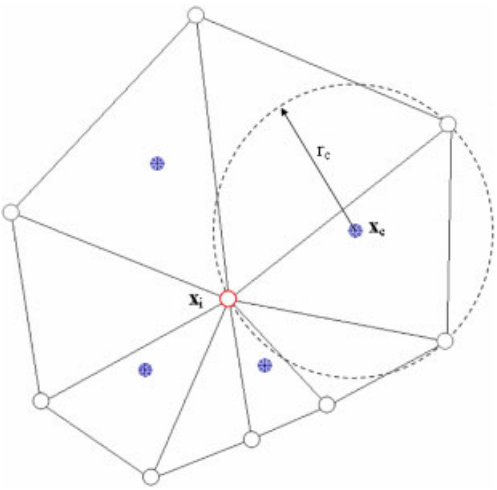

Figure 15. Refinement of a bi-dimensional cloud of points. The filled points $\mathbf{x}_c$ meet the requirements r_1–r_3 and, in consequence, are inserted around the star point $\mathbf{x}_i$.

$d_e = \max(|\mathbf{e}_j \cdot \hat{\mathbf{i}}|, |\mathbf{e}_j \cdot \hat{\mathbf{j}}|, |\mathbf{e}_j \cdot \hat{\mathbf{k}}|)$, where subscript j stands for each edge of the triangle/tetrahedron and $\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}}$ are unit vectors in each spatial direction.

$\mathbf{r}_3$. The distance from the candidate point $\mathbf{x}_c$ to another new point previously accepted is greater than the minimum admissible distance between points $r_{\min}$.

If any of the edges/triangles of the local Delaunay grid of nearest neighbours lies on the boundaries, a new candidate boundary point is obtained as an average of the position of the points defining this edge/triangle. The candidate boundary point is accepted if the distance to the nearest point is greater than $r_{\min}$. In our algorithm we perform the boundary refinement first and then we refine the discretization into the domain. Note that when the initial boundary discretization is very coarse, the straightforward procedure proposed for boundary refinement could deteriorate the boundaries, resulting in a lack of reliability of the computational model. In such cases, the position of new boundary points can be obtained using a higher-order interpolation of the underlying existing boundary points (cf. [33]). Figure 15 sketches the refinement procedure for a 2D cloud of points.

7.2.2. Removal of existing points. Point removal capabilities are indispensable for treating non-stationary problems. In this work, the removal of points is restricted only to the existing points that have been inserted in prior refinement levels. In other words, the initial set of points (original coarse discretization) is conserved through the calculation, though the spatial position of these points could change due to smoothing. This criterion avoids several time-consuming verifications and guarantees a minimum appropriate geometrical support for the calculation.

7.2.3. Update. Once the insertion and removal of points are finished, a few steps of a Laplacian smoothing are carried out on the affected area. This is particularly helpful when points have been removed in large quantities. After that, the clouds of points and shape functions concerning the new points are constructed. In addition, the data concerning existing clouds of points affected by

DOI: 10.1002/fld

the insertion of new points or smoothing are re-constructed. Finally, the flow variables at new points are calculated as an average of the variables at their previously existing nearest neighbours.

8. SOME EXAMPLES OF ADAPTIVE FP CALCULATIONS

In this section several numerical examples are presented in order to illustrate the performance of the proposed FP adaptive procedure. We begin with two computation cases intended to verify the adaptive numerical solution. The first example concerns a 2D adaptive calculation of a supersonic flow around a double-wedge airfoil and the second one deals with the solution of a shock-tube problem in a 2D domain. A third example is related to the solution of a transonic flow over an NACA 0012 airfoil and the fourth and last example involves a 3D flow calculation over the ONERA M6 wing. The two final calculation cases give an idea about the possibilities of application of the present adaptive FP meshless technique to practical engineering problems.

8.1. Supersonic flow past a double-wedge airfoil

This example resolves the flow around a double-wedge airfoil immersed in a supersonic flow. The airfoil has a unitary chord $c=1$ and the wedge angle is $\beta=20°$; the upstream Mach number is $M_\infty=2$ and the airfoil is set at an incidence angle $\alpha=0°$. The initial coarse discretization is composed by an unstructured distribution of 1279 points and second-order spatial approximations are built in clouds where $15\leqslant np\leqslant 20$. The final adapted discretization, achieved after 70 refinement levels, consists of 51 907 points. Next, the initial and the final adapted discretizations are shown in Figure 16.

Figure 17 presents a comparison between the analytical solution of the problem, calculated along an x-cut in the domain located 0.1c above the airfoil chord-line, and the numerical solution computed at successive refinement levels. How the numerical solution of successive refined-discretizations converges into the analytical solution of the problem can be observed. Finally, the time convergence of the problem is shown in Figure 18 where the complete process of the adaptive numerical computation can be seen. When the simulation starts, some time-steps are performed using the low-order scheme in order to initialize the flow field around the airfoil. Then, the flow solver switches to the high-order scheme and, even though it affects the convergence, the latter is recovered after a few time-steps. For a value of the density temporal residual of 1.0E−5, the first refinement level is performed. Then, consecutive refinement levels are carried out every 200 time-steps. Note that the peaks of the convergence curve correspond to each refinement level performed during the computation.

8.2. The shock-tube problem

The shock-tube problem is a one-dimensional non-stationary Riemann problem proposed by Sod in 1978 [34]. In this example we adopt a unitary-length bi-dimensional domain and carry out an adaptive shock-tube simulation defined by the following initial conditions:

$$\mathbf{U}(x,t_0)=\begin{cases}\mathbf{U}_\mathrm{L}=(1,0,0,2.5)^\mathrm{T}, & x\leqslant 0.5\\ \mathbf{U}_\mathrm{R}=(0.125,0,0,0.25)^\mathrm{T}, & x>0.5\end{cases} \tag{43}$$

DOI: 10.1002/fld

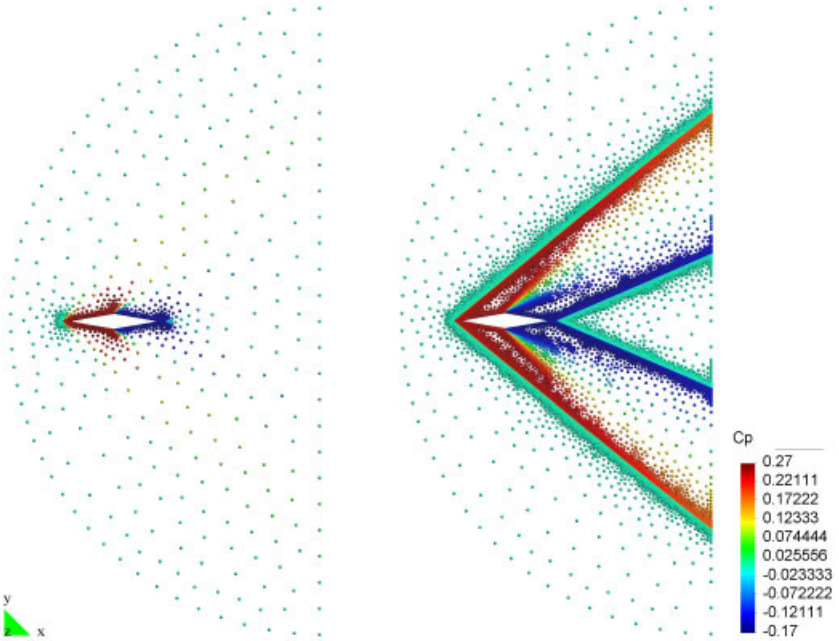

Figure 16. Supersonic flow past a double-wedge airfoil. Left: original coarse discretization and right: final adapted discretization (70 refinement levels). The coloured points show C_p results. $M_\infty = 2.0$ and $\alpha = 0°$.

which give a pressure ratio across the diaphragm $p_L/p_R = 10$ (notice that the diaphragm position is $x = 0.5$). According to the given initial conditions, the intensity of the shock is moderate and the flow regime after the expansion is subsonic.

The computational domain is initially discretized by a coarse homogeneous distribution of 217 points and second-order spatial approximations are calculated in clouds with $12 \leqslant np \leqslant 20$. After the rupture of the diaphragm, successive refinement levels are performed at regular periods. The simulation time in this example is $t = 0.2$ s, for which the adapted discretization reaches a total of 1761 points. Next, Figure 19 presents some snapshots of adapted discretizations taken at different times from the rupture of the diaphragm. There, the coloured points show flow density numerical results.

Figure 20 displays several comparisons between the numerical and the analytical solution for the density variable along the tube, corresponding to the simulation times pointed out in Figure 19.

In Figure 20, a considerable smoothing of the numerical solution can be observed in the first refinement levels ($t = 0.045$ and 0.1 s), for which the discontinuities are noticeably smeared. This fact can be explained to a great extend by the coarse discretization employed in order to start the simulation. Note that the number of points to be added in a given refinement level depends upon the flow field variables but also on the existing point discretization (cf. Section 7.2.1). Consequently, certain geometrical restrictions limit the maximum number of new points inserted at a given refinement level, which makes the discretization unable to adapt instantaneously to the flow variables in a proper manner. Nevertheless, a closer agreement between the numerical and the analytical solution is obtained for the simulation times $t = 0.14$, 0.19 and 0.20 s. In these cases, an improved flow resolution but also minor inaccuracies in the discontinuities location can be

DOI: 10.1002/fld

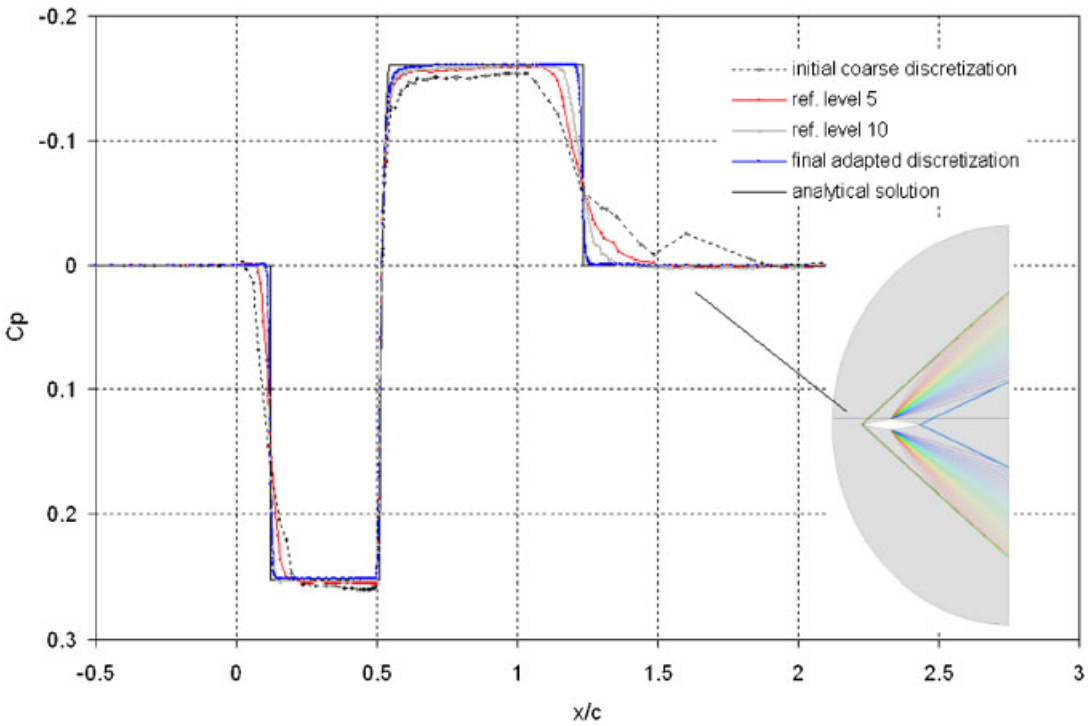

Figure 17. A comparison between the analytical C_p distribution along an x-cut on the domain and computed numerical results obtained at different refinement levels. The cut is located at $y/c=0.1$ and the airfoil leading edge coincides with the point $(x, y)=(0, 0)$. $M_\infty=2.0$ and $\alpha=0°$.

observed. We suspect that this behaviour could be related either to the straightforward procedure proposed to interpolate the numerical solution between the old and the new refined-discretization or to the Laplacian smoothing performed after each addition and/or removal of points. However, the solution should not be sensitive to the smoothing operations if a proper interpolation procedure is employed.

A numerical calculation performed with a fixed homogeneous discretization, having a point density similar to that in the final adapted discretization of Figure 19, is presented at the bottom right corner of Figure 20. Comparing the latter result with its counterpart obtained using the adaptive simulation, it is possible to observe that the numerical dissipation introduced by the refinement procedure is quite small. It should be noticed that the problem setting employed in both calculations is the same. Finally, it can be observed that the normalization adopted for calculating the refinement indicator may cause some detriment to the contact discontinuity resolution as stronger gradients are present at the shock location. In cases like this, it would be useful to adopt a local criterion for calculating the refinement indicator.

8.3. Transonic flow around an NACA 0012 airfoil

This example concerns the computation of a transonic inviscid flow past an NACA 0012 airfoil. The freestream Mach number is $M_\infty=0.8$ and the incidence angle is $\alpha=1.25°$. The initial spatial

DOI: 10.1002/fld

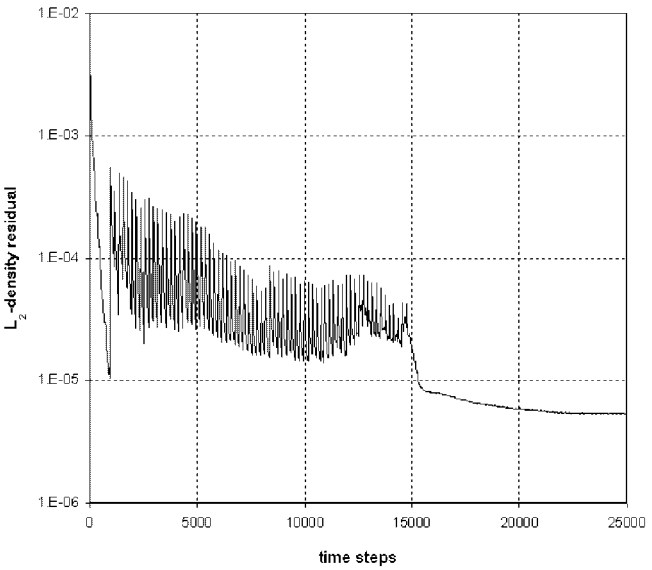

Figure 18. Convergence history of the double-wedge airfoil calculation (70 refinement levels). $M_\infty = 2.0$ and $\alpha = 0.0^\circ$.

discretization involves an unstructured distribution of 976 points and second-order spatial approximations are calculated in clouds with $15 \leqslant np \leqslant 20$. The finest adapted discretization consists of 4938 points and is achieved after 15 refinement levels. Both, the initial and the final discretizations are shown in Figures 21 and 22 respectively.

Notice that the adaptive procedure captures all the flow features with precision. The strong shock wave on the upper side of the airfoil, the weaker shock on its lower side and the leading and trailing edge regions are appropriately captured via the refinement procedure. Figure 23 shows the C_p field around the airfoil calculated for the final adapted discretization.

The computed C_p distribution on the airfoil is compared with numerical reference results [35] in Figure 24, where good agreement can be observed. Finally, the time convergence history of the problem is presented in Figure 25.

8.4. A 3D example: the ONERA M6 wing

This example solves the 3D flow around the ONERA M6 wing adopting the freestream conditions given in Section 6.2. The initial coarse discretization consists of an unstructured distribution of 66 864 points and second-order approximation bases are employed in clouds with $30 < np < 45$. In this simulation the adapted discretization reaches a total of 102 592 points after 35 refinement levels. Next, Figure 26 shows the original and final discretizations of the wing; coloured points display C_p results.

The initial discretization of the wing consists of 14 221 points and 28 314 triangle elements, whereas the final adapted discretization is composed of 15 537 points and 30 942 triangles. Note that new points are mainly concentrated around the strong shock wave spanning the wing where large gradients are detected. In order to make the refinement indicator (42) also sensitive

DOI: 10.1002/fld

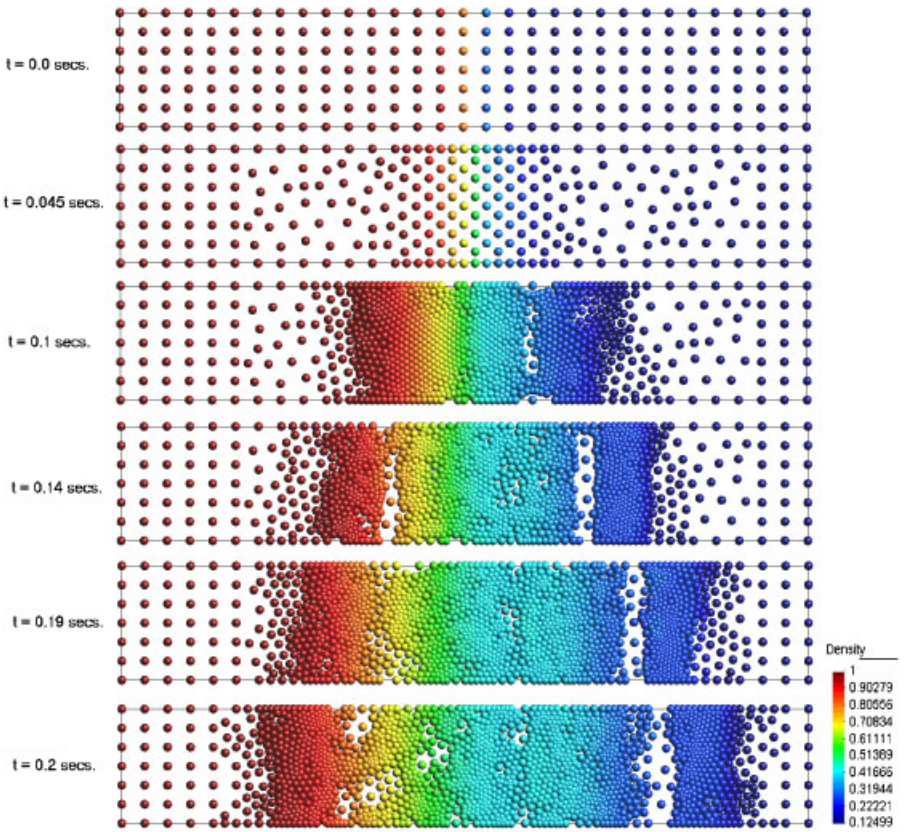

Figure 19. Adapted discretizations obtained for the shock-tube problem ($p_L/p_R = 10$) at different times from the rupture of the diaphragm (the top image shows the initial coarse discretization).

to the smaller gradients in the flow field, we can decrease the parameter $\varphi_{\max}$ or change the normalization criterion. However, as the indicator becomes more sensitive, the refinement procedure loses its local character. This would lead to an insertion of large sets of new points for each refinement level and the convergence of the problem could be seriously affected in some cases. Thus, the adoption of local maxima for normalizing the indicator seems a more adequate choice.

Figure 27 compares the C_p distributions along two sections of the wing calculated with the original and the finest discretization. In the same figure, a view of the finest adapted point discretization for a cut in the plane x–z of the domain (passing through the same spanwise stations) is presented. Finally, the convergence history of the problem is shown in Figure 28.

Regarding the computational cost of the proposed FP adaptive technique, numerical experiments show that the CPU-time required by each refinement level is approximately equal to the time involved for the update stage (cf. Section 7.2.3) and the cost of inserting and removing points is almost negligible. In general, the overall CPU-time involved for each refinement level is only a fraction of the time required for advancing the problem solution a single time-step.

DOI: 10.1002/fld

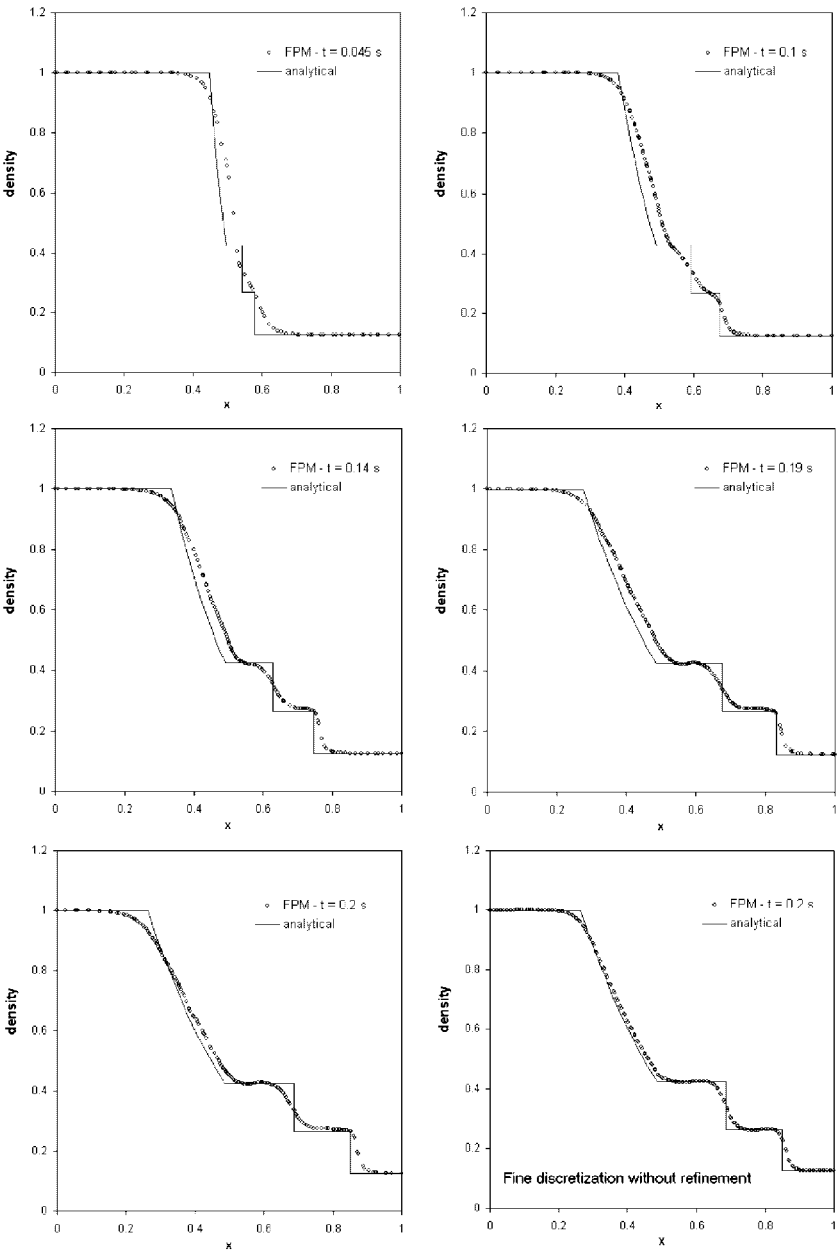

Figure 20. Comparison between numerical and analytical solutions for the density distribution along the centreline of the shock-tube at different times from the rupture of the diaphragm ($p_L/p_R = 10$). The numerical solution at the bottom right corner is calculated using a fine discretization without performing any refinement level.

DOI: 10.1002/fld

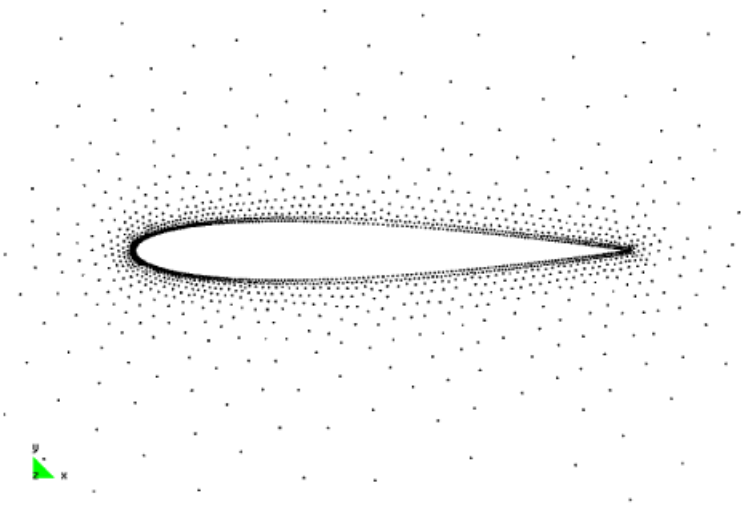

Figure 21. A view of the original coarse discretization in the proximity of the NACA 0012 airfoil.

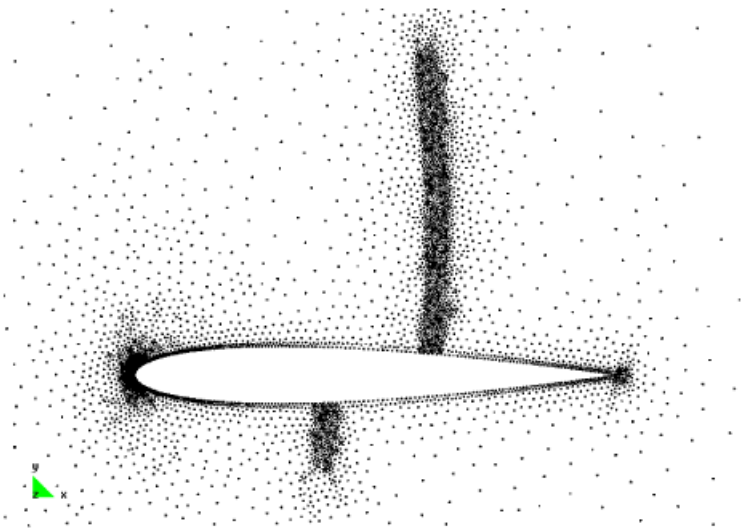

Figure 22. A view of the finest adapted discretization in the proximity of the NACA 0012 airfoil obtained after 15 refinement levels.

9. CONCLUSIONS

An adaptive finite point method (FPM) for compressible flow calculations has been presented. On the basis of a robust WLSQ procedure and an iteratively improved local approximation, an upwind semi-discrete scheme is constructed for each cloud of points. This methodology, in conjunction with a multi-stage time integration scheme, allows solving real 3D problems minimizing the

DOI: 10.1002/fld

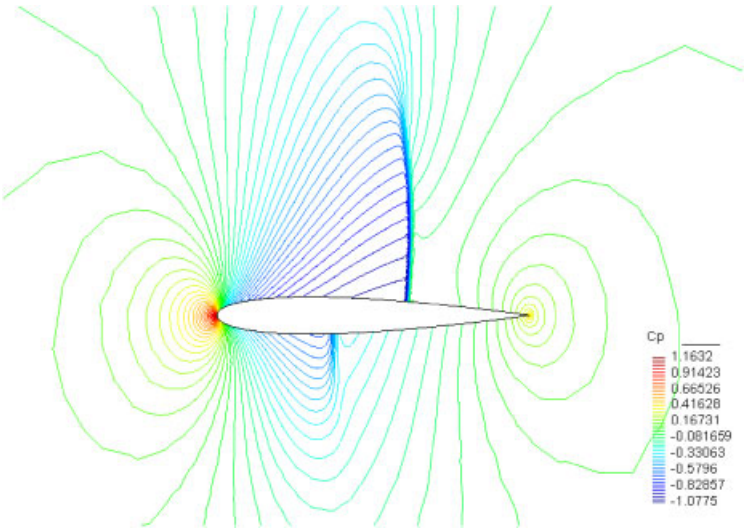

Figure 23. C_p isolines in the near field of the NACA 0012 airfoil obtained with the finest adapted discretization. $M_\infty = 0.80$ and $\alpha = 1.25°$.

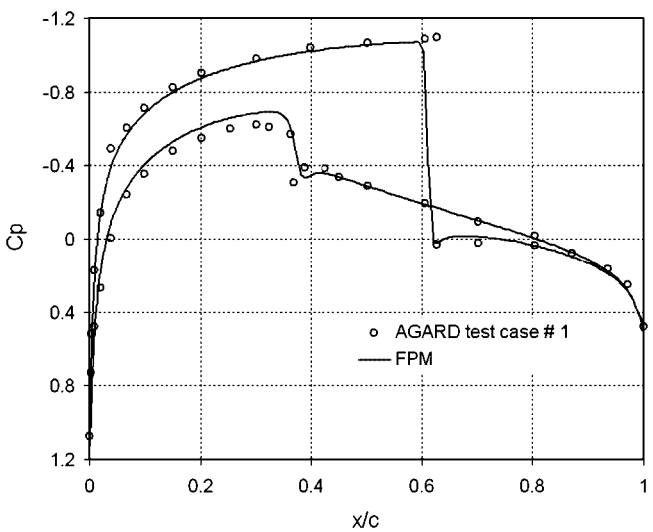

Figure 24. C_p distribution on the NACA 0012 airfoil obtained with the finest adapted discretization. A comparison between computed and numerical reference results [35]. $M_\infty = 0.80$ and $\alpha = 1.25°$.

dependence of the numerical results on the spatial discretization of the analysis domain, the local cloud topology and the parameters of the local approximation. All these are important achievements, which make possible further enhancement and extension of the FPM capabilities for practical 3D applications.

DOI: 10.1002/fld

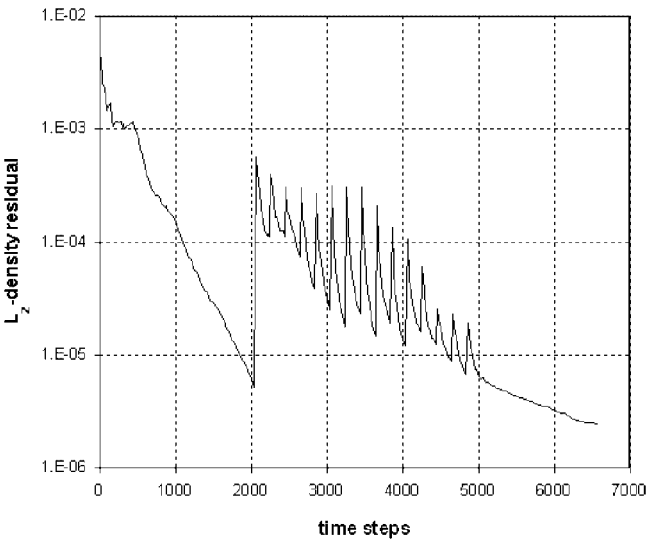

Figure 25. Convergence history of the NACA 0012 airfoil calculation (15 refinement levels). $M_\infty = 0.80$ and $\alpha = 1.25°$.

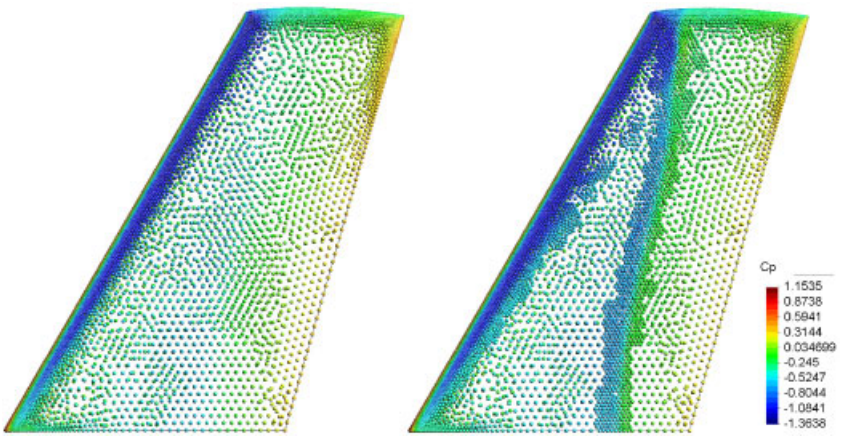

Figure 26. A view of the upper side of the ONERA M6 wing. Left: original coarse discretization. Right: finest adapted discretization (35 refinement levels). $M_\infty = 0.84$ and $\alpha = 3.06°$.

In the introduction to this article we made reference to certain topics in numerical simulation, which offer good opportunities for the development and promotion of meshless techniques. With the aim of exploiting these opportunities, an adaptive FPM for compressible flow calculations has been developed. Several test cases involving stationary and non-stationary flow problems have been presented with the purpose of exemplifying the performance of the proposed technique. All the examples demonstrate that an adaptive FPM is capable of properly resolve the essential flow features, achieving robust and reliable adaptive solutions with a low computational cost. Although

DOI: 10.1002/fld

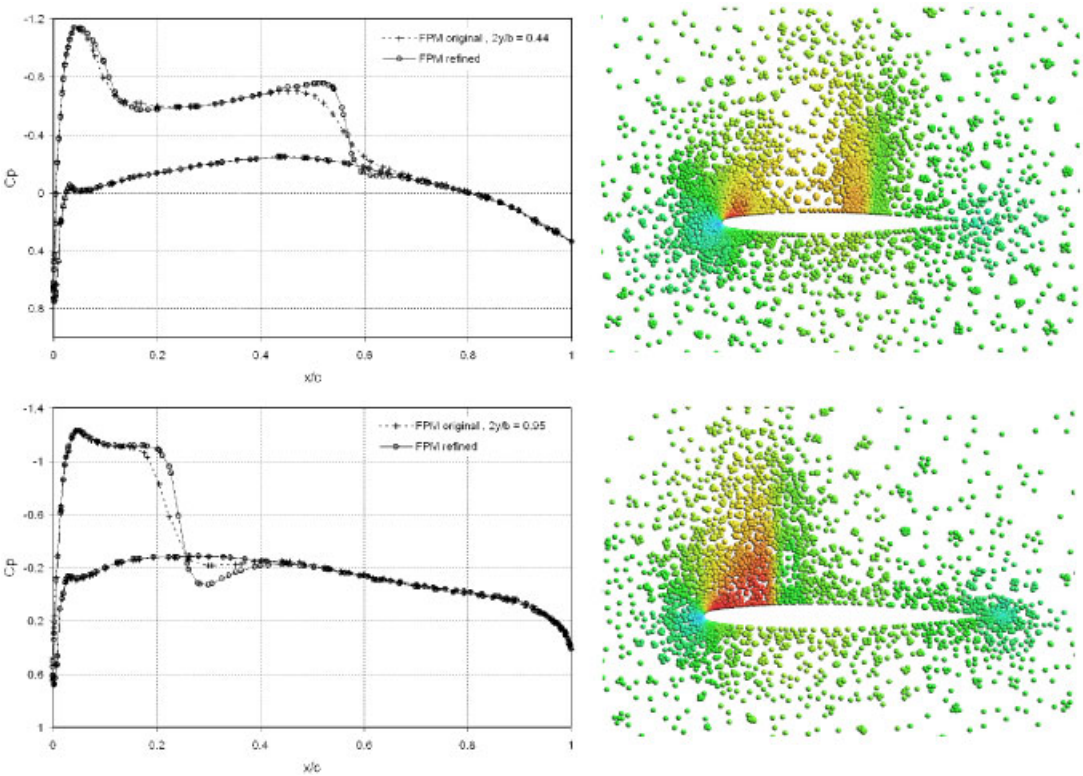

Figure 27. Left: C_p distributions along two wing sections $\eta=0.44$ (top) and $\eta=0.95$ (bottom) calculated with the original and the final adapted discretization. Right: cuts x–z of the finest refined domain passing through wing stations $\eta=0.44$ (top) and $\eta=0.95$ (bottom). ONERA M6 wing $M_\infty=0.84$ and $\alpha=3.06^\circ$.

some numerical tests (of which a few have been reported here) highlight the need for more accurate refinement criteria and an improved treatment of moving discontinuities, the overall performance of the proposed adaptive FPM is highly satisfactory and this can be seen as the main achievement of this work.

Real viscous flow involves certain features where meshless techniques, and especially adaptive meshless techniques, could make important contributions, e.g. boundary layer discretization and shock-boundary layer interaction problems. In this sense, we have developed the basic tools for tackling these kinds of problems and solving them constitutes our next short-term goal.

Regarding computational efficiency we must say that at present we still lack precise performance comparisons between the FPM described here and conventional discretization techniques. However, we estimate that the computational cost of a 3D FP computation using the methodology presented in this paper would typically exceed a similar FE-based computation by a cost factor of 3 being 5 a typical value. Hence, if a competitive FPM is to be achieved, an improvement in computational efficiency is indispensable. In that respect, numerous techniques can be implemented in order to accelerate convergence to the steady state. Combining these techniques with a suitable data structure and an optimized way to perform the numerical calculations, it is possible to enhance

DOI: 10.1002/fld

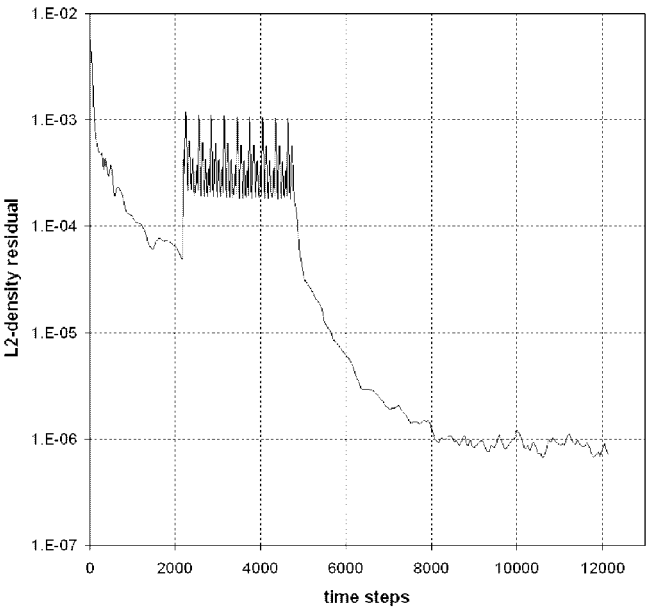

Figure 28. Convergence history of the ONERA M6 wing adaptive calculation (35 refinement levels). $M_\infty = 0.84$ and $\alpha = 3.06°$.

the efficiency of the present FPM considerably. Moreover, performance comparisons between the present FPM and other meshless techniques accomplishing similar tasks are essential for placing the FPM into the actual meshless methods scenario. In conclusion, the results obtained with the FPM are very encouraging, though efficiency is still a pending matter. Consequently, future research efforts will take highly into consideration the improvement of this key point.

ACKNOWLEDGEMENTS

The first author would like to acknowledge the support of Alßan Program, the European Union Programme of High Level Scholarships for Latin America, scholarship No. E04D027284AR. We express our gratitude to Dr Nestor Calvo for helping with the point generation technique employed in this work. Last but not least, the many valuable contributions of Dr Roberto Flores to this work are also gratefully acknowledged.

REFERENCES

1. Belytschko T, Krongauz Y, Organ D, Fleming M, Krysl P. Meshless methods: an overview and recent developments. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**:3–47.
2. Fries T, Matthies H. *Classification and Overview of Meshfree Methods*. Department of Mathematics and Computer Science, Technical University of Braunschweig. Inf. 2003-3, 2004.
3. Li S, Liu WK. Meshfree and particle methods and their applications. *Applied Mechanics Reviews* 2002; **55**:1–34.
4. Gu YT. Meshfree methods and their comparisons. *International Journal of Computational Methods* 2005; **4**: 477–515.
5. Duarte CA. A review of some meshless methods to solve partial differential equations. *TICAM Report 95-06*, Texas Institute for Computational and Applied Mathematics, 1995.

DOI: 10.1002/fld

6. Liu WK, Chen Y, Jun S, Belytschko T. Overview and application of the reproducing Kernel particle method. *Archives of Computational Methods in Engineering* 1996; **5**(1):3–80.
7. Dolbow J, Belytschko T. An introduction to programming the meshless element free-Galerkin method. *Archives of Computational Methods in Engineering* 1998; **5**(3):207–241.
8. Oñate E, Idelsohn S, Zienkiewicz OC, Taylor RL, Sacco C. A finite point method for analysis of fluid mechanics problems. Applications to convective transport and fluid flow. *International Journal for Numerical Methods in Engineering* 1996; **39**:3839–3866.
9. Oñate E, Idelsohn S, Zienkiewicz OC, Fisher T. A finite point method for analysis of fluid flow problems. *Proceedings of the 9th International Conference on Finite Elements Methods in Fluids*, Venice, Italy, 1995; 15–21.
10. Oñate E, Idelsohn S, Zienkiewicz OC, Taylor RL, Sacco C. A stabilized finite point method for analysis of fluid mechanics problems. *Computer Methods in Applied Mechanics and Engineering* 1996; **139**:315–346.
11. Oñate E, Sacco C, Idelsohn S. A finite point method for incompressible flow problems. *Computing and Visualization in Science* 2000; **3**:67–75.
12. Fischer TR. A contribution to adaptive numerical solution of compressible flow problems. *Ph.D. Thesis*, Universitat Politècnica de Catalunya, 1996.
13. Sacco C. Development of the finite point method in fluid mechanics (in Spanish). *Ph.D. Thesis*, School of Civil Engineering, Universitat Politècnica de Catalunya, 2001.
14. Löhner R, Sacco C, Oñate E, Idelsohn S. A finite point method for compressible flow. *International Journal for Numerical Methods in Engineering* 2002; **53**:1765–1779.
15. Oñate E, Perazzo F, Miquel J. A finite point method for elasticity problems. *Computers and Structures* 2001; **79**:2151–2163.
16. Perazzo F, Miquel J, Oñate E. A finite point method for solids dynamic problems (in Spanish). *Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería* 2004; **20**(3):235–246.
17. Perazzo F, Oller S, Miquel J, Oñate E. Advances in the finite point method for solid mechanics (in Spanish). *Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería* 2006; **22**(3):153–167.
18. Oñate E. Derivation of stabilized equations for numerical solution of advective–diffusive transport and fluid flow problems. *Computer Methods in Applied Mechanics and Engineering* 1998; **151**:233–265.
19. Sridar D, Balakrishnan N. An upwind finite difference scheme for meshless solvers. *Journal of Computational Physics* 2003; **189**:1–29.
20. Praveen C. A positive meshless method for hyperbolic equations. *Report 2004-FM-16*, ARDB Centre of Excellence for Aerospace CFD, Department of Aerospace Engineering, Indian Institute of Science, 2004.
21. Boroomand B, Tabatabaei AA, Oñate E. Simple modifications for stabilization of the finite point method. *International Journal for Numerical Methods in Engineering* 2005; **63**:351–379.
22. Ortega E, Oñate E, Idelsohn S. An improved finite point method for three-dimensional potential flows. *Computational Mechanics* 2007; **40**:949–963.
23. Perazzo F, Löhner R, Perez-Pozo L. Adaptive methodology for meshless finite point method. *Advances in Engineering Software* 2007; **39**:156–166.
24. Lancaster P, Salkauskas K. Surfaces generated by moving least squares methods. *Mathematics of Computation* 1981; **37**:141–158.
25. Demmel JW. *Applied Numerical Linear Algebra*. Society for Industrial and Applied Mathematics: Philadelphia, 1997.
26. Idelsohn S, Calvo N, Oñate E. Polyhedrization of an arbitrary 3D point set. *Computer Methods in Applied Mechanics and Engineering* 2003; **192**:2649–2667.
27. Roe PL. Approximate Riemann solvers, parameter vectors and difference schemes. *Journal of Computational Physics* 1981; **43**:357–372.
28. Turkel E. Improving the accuracy of central difference schemes. *ICASE Report 88-53*, 1988.
29. Van Leer B. Towards the ultimate conservative difference scheme. V-A second-order sequel to Godunov's method. *Journal of Computational Physics* 1979; **32**:101–136.
30. Schmitt V, Charpin F. Pressure distributions on the ONERA-M6-wing at transonic Mach numbers. Experimental data base for computer program assessment. *Report of the Fluid Dynamics Panel Working Group 04*, AGARD AR 138, 1979.
31. NPARC Alliance CFD verification and validation web site. Web page: http://www.grc.nasa.gov/WWW/wind/valid/archive.html (26 October 2007).

DOI: 10.1002/fld

32. Loving D, Estabrooks B. Transonic-wing investigation in the Langley 8-foot high-speed tunnel at high subsonic Mach numbers and at a Mach number of 1.2. Analysis of pressure distribution of wing-fuselage configuration having a wing of 45° sweepback, aspect ratio 4, taper ratio 0.6, and NACA 65A006 airfoil section. *Research Memorandum NACA RM L51F07*, National Advisory Committee for Aeronautics, 1951.
33. Zorin D, Schröder P, Sweldens W. Interpolating subdivision for meshes with arbitrary topology. *Proceedings of SIG-GRAPH* 1996; **96**:189–192.
34. Sod GA. A survey of several finite-differences methods for systems of nonlinear hyperbolic conservation laws. *Journal of Computational Physics* 1978; **27**:1–31.
35. Pulliam TH, Barton JT. Euler computations of AGARD Working Group 07 Airfoil Test Cases. *AIAA 23rd Aerospace Summer Meeting*, Reno, NE, AIAA Paper 85-0018, 1985.

DOI: 10.1002/fld